TUGAS AKHIR – TE 141599

SISTEM PEMANDU WISATA DIGITAL DI KEBUN BINATANG

Reza Zhafiri
NRP 2211100037

Dosen Pembimbing
Ronny Mardiyanto, ST., MT., Ph.D
Dr. Ir. Djoko Purwanto, M.Eng.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2015

FINAL PROJECT – TE 141599

DIGITAL TOUR GUIDE SYSTEM IN ZOO

Reza Zhafiri
NRP 2211100037

Supervisor
Ronny Mardiyanto, ST., MT., Ph.D
Dr. Ir. Djoko Purwanto, M.Eng.

DEPARTEMENT OF ELECTRICAL ENGINEERING
Faculty of Industrial Technology
Sepuluh Nopember Institute of Technology
Surabaya 2015

SISTEM PEMANDU WISATA DIGITAL DI KEBUN BINATANG

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Elektronika
Jurusan Teknik Elektro
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember

Menyetujui:

Dosen Pembimbing I	Dosen Pembimbing II
Ronny Mardiyanto, ST., MT., Ph.D	Dr. Ir. Djoko Purwanto, M.Eng.
NIP: 198101182003121003	NIP: 196512111990021002

SURABAYA
JULI, 2015

SISTEM PEMANDU WISATA DIGITAL DI KEBUN BINATANG

Nama : Reza Zhafiri
NRP : 2211100037
Jurusan : Teknik Elektro
Dosen Pembimbing : 1. Ronny Mardiyanto, ST., MT., Ph.D
2. Dr. Ir. Djoko Purwanto, M.Eng.

ABSTRAK

Kebun Binatang merupakan tempat wisata edukasi yang memiliki tujuan meningkatkan wawasan masyarakat mengenai satwa iklim tropis. Untuk mendukung tercapainya tujuan tersebut dibutuhkan pemandu wisata yang dapat menjelaskan informasi mengenai satwa yang ada secara jelas dan lengkap. Namun, tidak banyak penunjung yang menggunakan fasilitas pemandu wisata sehingga informasi yang didapatkan oleh wisatawan kurang lengkap. Oleh karena itu, sebuah teknologi pemandu wisata digital dikembangkan untuk mengatasi permasalahan tersebut.

Perangkat pemandu wisata digital ini memungkinkan pengunjung untuk mendapatkan penjelasan setiap satwa melalui media video dan audio dengan lebih menarik dan lebih lengkap. Pemandu wisata digital ini merupakan perangkat berbentuk *Head-mounted Device* sehingga memudahkan pengunjung ketika memakainya. Perangkat ini terdiri dari raspberry pi sebagai bagian utamanya dan sebuah kamera untuk menangkap gambar *marker* di setiap wilayah satwa. Dengan menggunakan *image processing*, ketika kamera mendeteksi marker maka perangkat akan menampilkan video informasi secara otomatis.

Perangkat pemandu wisata ini dapat menampilkan video informasi penjelasan secara otomatis dalam kondisi optimal. Hasil dari perangkat pada pembuatan tugas akhir ini dapat memproses gambar marker pada kecepatan hinga 7fps dan kondisi cahaya minimal 10 lux. Perangkat ini juga memiliki posisi optimal dengan jarak maksimal 1.5 meter dan sudut maksimal 70 derajat terhadap gambar marker.

Penggunaan pemandu wisata digital belum banyak dikembangkan di Indonesia. Oleh karena itu, perangkat ini diharapkan mampu menambah wawasan wisatawan tentang satwa yang ada di Kebun Binatang tersebut dan akhirnya dapat menarik wisatawan sehingga dapat mengembangkan objek wisata tersebut.

Kata kunci : *Image Processing*, Raspberry Pi, *Head Mounted Display*

DIGITAL TOUR GUIDE SYSTEM IN ZOO

Name : Reza Zhafiri
NRP : 2211100037
Department : Electrical Engineering
SuperVisor : 1. Ronny Mardiyanto, ST., MT., Ph.D
2. Dr. Ir. Djoko Purwanto, M.Eng.

ABSTRACT

Zoo is an educational tourism object about animal diversity. However, decreasing development of Indonesian tourism industry recently has made zoo uninteresting tourism destination to visit although a zoo can become an important learning media for society. To improve society's knowledge about existing tropical wildlife, a tour guide who can explain the information about the animals existed clearly and completely is required. However, not many visitors use the tour guide facility so that the information obtained is not complete. Therefore, a Digital Tour Guide technology has been developed to solve these problems.

This Digital Tourist Guide device allows visitors to get an explanation of each animal through the video and audio more appealingly and completely. It consists of shaped glasses using a special LCD display to produce an image that can be seen directly by visitors wearing them. This device consists of a raspberry pi as the main part and a camera to capture images of marker in each wildlife region. By using image processing on the raspberry pi, whenever the camera captures the image marker in each animal cage, it will automatically present the video containing an explanation of these animals.

This device would display the information video automatically in optimal condition. The device in this final project could process images up to 7 fps, with at least 10 lux light condition. This device also work optimum in limited position, which up to 1.5 meters and 70 degree from marker image.

The use of a Digital Tour Guide has not been developed in Indonesia. Therefore, this technology is expected to improve traveler insight about animals living in zoo and finally can attract more visitors which can develop the tourism object itself.

Keywords : Image Processing, Raspberry Pi, Head Mounted Display

KATA PENGANTAR

Segala puji syukur bagi Allah SWT atas kemudahan, kesehatan, dan kesempatan yang telah diberikan dari awal hingga akhir pengerjaan tugas akhir berjudul:

Sistem Pemandu Wisata Digital di Kebun Binatang

Penyusunan Tugas Akhir ini adalah sebagai salah satu persyaratan untuk menyelesaikan pendidikan S1 pada bidang studi Elektronika Industri di Jurusan Teknik Elektro ITS Surabaya dan diharapkan mampu menambah wawasan dalam bidang keahlian elektronika.

Dalam penyusunan Tugas Akhir ini dan selama studi di Institut Teknologi Sepuluh Nopember, penulis mendapatkan bantuan, bimbingan, dan dukungan tak ternilai dari berbagai pihak. Untuk itu penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Kedua orangtua saya beserta adik-adik saya atas dukungan dan motivasi tanpa henti selama waktu pengerjaan tugas akhir ini.
2. Bapak Ronny Mardiyanto, ST., MT., Ph.D selaku dosen pembimbing pertama, atas kesediaannya memberikan masukan dan waktu untuk menyelesaikan tugas akhir ini.
3. Bapak dosen Dr. Ir. Djoko Purwanto, M.Eng. selaku pembimbing kedua, atas bimbingan dan penjelasan yang diberikan.
4. Seluruh Dosen Bidang Studi Elektronika Jurusan Teknik Elektro ITS, atas bimbingan serta ilmu perkuliahan yang telah diberikan.
5. Seluruh rekan – rekan anggota labolatorium elektronika atas kebersamaan dan kerjasamanya selama ini.
6. Seluruh keluarga besar Teknik Elektro ITS, para dosen, karyawan, mahasiswa atas dukungan, masukan, dan kerjasamanya selama masa kuliah dan proses pengerjaan tugas akhir.
7. Semua pihak yang telah membantu dalam pelaksanaan dan penyusunan laporan tugas akhir yang tidak dapat penulis sebutkan satu per satu.

Besar harapan penulis agar tugas akhir ini dapat memberikan manfaat dan masukan bagi banyak pihak. Oleh karena itu penulis mengharapkan kritik dan saran dari pembaca yang bersifat membangun untuk pengembangan kearah yang lebih baik.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1. Latar Belakang

Kebun binatang merupakan tempat wisata edukasi yang berisi berbagai macam satwa. Salah satu kebun binatang terbesar yang terdapat di Indonesia adalah Kebun Binatang Surabaya. Kebun Binatang Surabaya merupakan salah satu tempat wisata kebun binatang yang sudah cukup lama berdiri dan telah menjadi salah satu ikon wisata kota Surabaya. Sebagai tempat wisata edukasi, kebun binatang terbesar se-Asia Tenggara dengan jumlah satwa hingga lebih dari 4000 ekor ini bertujuan untuk meningkatkan wawasan masyarakat mengenai satwa iklim tropis yang ada. Oleh karena itu, dibutuhkan pemandu wisata yang mampu memberikan informasi mengenai satwa yang ada secara lengkap dan jelas.

Namun demikian, pada umumnya tidak banyak masyarakat yang menggunakan jasa pemandu wisata ketika mengunjungi Kebun Binatang tersebut. Hal ini disebabkan oleh biaya pengunaan jasa pemandu wisata yang cukup mahal. Di sisi lain, fasilitas papan informasi pada kebun binatang kurang diperhatikan oleh pihak pengelola kebun binatang sehingga papan tersebut banyak terdapat dalam keadaan rusak. Kondisi tersebut mengakibatkan pengunjung tidak tertarik untuk membaca informasi mengenai satwa sehingga fungsi pembelajaran kebun binatang untuk masyarakat tidak tercapai.

Untuk mengatasi permasalahan tersebut, dibutuhkan fasilitas pendukung untuk membantu pengunjung mendapatkan informasi yang lengkap dan jelas mengenai satwa yang terdapat di kebun binatang sehingga dapat mengoptimalkan fungsi kebun binatang sebagai sarana edukasi masyarakat. Fasilitas pendukung yang diperlukan tentu harus dapat memberikan informasi mengenai satwa secara lengkap dan jelas. Di samping itu, fasilitas ini juga harus mampu memberikan penjelasan secara menarik sehingga dapat menarik lebih banyak wisatawan.

Salah satu fasilitas yang dapat diaplikasikan pada tempat wisata tersebut adalah perangkat pemandu wisata digital dengan memanfaatkan teknologi *wearable augmented reality*. Perangkat ini memungkinkan penggunanya untuk dapat merasakan pemandu wisata secara visual melalui media audio video secara interaktif. Pada akhirnya, fasilitas ini diharapkan mampu menambah wawasan wisatawan tentang satwa yang

ada di Kebun Binatang secara menarik dan akhirnya dapat meningkatkan jumlah wisatawan sehingga dapat mengembangkan objek wisata tersebut.

1.2. Perumusan Masalah

Permasalahan yang akan dibahas dalam tugas akhir ini antara lain:

1. Bagaimana membuat sistem pemandu wisata digital dengan perangkat *wearable augmented reality* yang sesuai untuk tempat pariwisata Kebun Binatang
2. Bagaimana membuat tampilan antar-muka pemandu wisata digital dengan pemutar media interaktif dan pengolahan citra pada Raspberry-pi
3. Bagaimana mendeteksi gambar *marker* dengan menggunakan pengolahan citra OpenCV pada Raspberry-pi
4. Bagaimana performa hasil proses pencitraan gambar *marker* oleh Open-CV pada Raspberry-pi

1.3. Batasan Permasalahan

Batasan permasalahan dan asumsi yang digunakan dalam penyelesaian Tugas Akhir ini antara lain:

1. Perangkat utama yang digunakan adalah *single board computer* berupa Raspberry-pi jenis B+
2. Kamera yang digunakan adalah kamera USB-webcam dengan resolusi 5MP
3. Cuaca pada saat kondisi pengujian tidak sedang dalam kondisi hujan, mendung, atau pada malam hari
4. Seluruh data audio visual yang digunakan sudah terdapat di dalam memori Raspberry-pi
5. Daya tahan baterai pada perangkat hanya dapat bertahan hingga 1 jam
6. Ukuran alat disesuaikan dengan ukuran pembuat

1.4. Tujuan Penelitian

Tujuan yang ingin dicapai dalam tugas akhir ini adalah sebagai berikut:

1. Memperoleh perangkat yang tepat sebagai media audio visual pada tempat pariwisata outdoor
2. Memperoleh teknik pencitraan gambar dan tampilan *audio visual* yang tepat untuk dapat dijalankan dengan melalui Raspberry-pi

3. Mewujudkan sebuah sistem pemandu wisata digital yang dapat diimplementasikan pada tempat pariwisata Kebun Binatang.

1.5. Metode Penelitian

Metode penelitian yang digunakan dalam penyelesaian tugas akhir ini antara lain:

1. Studi Literatur
   Studi literatur dilakukan dengan mengumpulkan dasar teori yang dibutuhkan untuk mendukung penulisan tugas akhir ini dari berbagai jurnal, buku, dan artikel di internet yang meliputi:
   - Teknik pembacaan gambar *marker* pada OpenCV
   - Penggunaan tampilan audio visual pada Raspberry-pi secara terkontrol melalui koneksi *dbus*
   - Perangkat pemandu wisata digital yang sudah ada sebagai referensi dan pembanding
   - Cara mempercepat proses kerja OpenCV pada Raspberry-pi
2. Perancangan Hardware
   Perancangan hardware pada penulisan tugas akhir ini berupa *head-mounted device* yang dapat dipakai langsung layaknya kacamata, yang meliputi:
   - Peletakan layar LCD yang disesuaikan dengan lensa pada perangkat wearable *augmented reality*
   - Pemilihan lensa dengan titik focus yang sesuai dalam memantulkan tampilan layar LCD hingga ke pengguna
   - Pemasangan kamera yang sesuai pada perangkat wearable *augmented reality*
3. Perancangan *Software*
   Perancangan *software* pada penulisan tugas akhir ini terbagi menjadi 2 bagian, yaitu:
   - Perancangan program untuk pengolahan citra gambar *marker* pada OpenCV, serta perancangan bentuk gambar *marker* yang akan dideteksi
   - Perancangan program untuk pengintegrasian antara *software* pemutar media pada Raspberry-pi dengan OpenCV
4. Pengujian Sistem

Pengujian sistem dilakukan dengan menguji pendeteksian gambar *marker* melalui kamera oleh OpenCV, dan kesesuaian tampilan media audio visual yang dihasilkan oleh pemutar media pada Raspberry-pi. Pengujian dilakukan dengan mengubah-ubah beberapa variabel untuk mengetahui tingkat kehandalan dari perangkat tersebut.

5. Penulisan Buku
   Penulisan buku dilakukan sebagai laporan tugas akhir, dan dilakukan setelah didapatkan data-data hasil yang dibutuhkan.

1.6. Sistematika Penulisan

Untuk mempermudah pembahasan, dalam penulisan tugas akhir ini dibagi menjadi lima bab dan beberapa subbab dengan sistematika sebagai berikut:

Bab 1: Pendahuluan
Bab ini membahas mengenai latar belakang, rumusan masalah, tujuan, batasan masalah, sistematika penulisan, metodologi, dan relevansi.

Bab 2: Dasar Teori
Bab ini menjelaskan tentang bahan-bahan teori penunjang dalam pengerjaan tugas akhir ini, diantaranya: Pendeteksian gambar *marker* melalui OpenCV, media pemutar audio visual pada Raspberry-pi.

Bab 3: Perancangan Sistem
Bab ini menjelaskan tentang langkah-langkah yang dilakukan untuk merancang hingga membangun sistem dalam *hardware* maupun *software*.

Bab 4: Pengujian dan Analisis
Bab ini menjelaskan tentang hasil pengujian dan analisa yang telah dilakukan dari perangkat yang telah dibuat.

Bab 5: Penutup
Bab ini merupakan bagian akhir dari penulisan buku tugas akhir ini yang berisi kesimpulan serta saran untuk pengembangan selanjutnya.

1.7. Relevansi

Hasil dari tugas akhir ini diharapkan mampu memandu pengunjung dalam berpariwisata mengunjungi Kebun Binatang. Selain itu juga diharapkan perangkat dari hasil tugas akhir ini dapat menjadi pemandu wisata yang mampu membantu pengunjung Kebun Binatang untuk

belajar dan mengetahui informasi mengenai satwa yang berada di Kebun Binatang dengan lebih jelas.

BAB 2
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. Pemandu Wisata

Menurut kamus besar bahasa Indonesia, pemandu wisata atau pramuwisata adalah orang yang pekerjaannya mendampingi wisatawan dengan mengatur perjalanan dan memberi penjelasan tentang tempat yang dikunjungi, atau sebagai orang yang bertugas memandu wisatawan. [1]

Pemandu wisata sangat dibutuhkan untuk memandu wisatawan dalam mengunjungi tempat wisata, khususnya pada wisata edukasi seperti museum, kebun binatang, dan sebagainya. Pada tempat wisata edukasi tersebut, pada umumnya memiliki tujuan untuk menambah wawasan pengunjung mengenai objek wisata yang ada di dalamnya. Contohnya yaitu pada museum tugu pahlawan, memiliki tujuan wisata agar pengunjung dapat lebih memahami sejarah peristiwa perang di Surabaya. Contoh lain juga terdapat di Kebun Binatang Surabaya, yang memiliki tujuan wisata agar pengunjung dapat lebih mengetahui informasi tentang beraneka ragam satwa.

Dengan adanya pemandu wisata, maka wisatawan dapat lebih banyak menerima informasi terkait dengan objek yang terdapat pada tempat wisata tersebut. Meski demikian, banyak wisatawan yang masih jarang menggunakan jasa para pemandu wisata. Sehingga, informasi yang didapatkan oleh wisatawan dari tempat wisata yang dikunjunginya tidak terlalu lengkap.

2.2. Kebun Binatang Surabaya

Kebun Binatang Surabaya (KBS) berlokasi di Surabaya Selatan adalah salah satu kebun binatang yang populer di Indonesia, terletak di Jalan Setail No. 1 Surabaya. KBS memiliki berbagai jenis binatang tropis. Selain itu terdapat pula Aquarium, karantina toxidemi dan ruang nokturama (binatang malam). KBS merupakan kebun binatang yang terbesar di Asia Tenggara. Didalamnya terdapat lebih dari 300 spesies satwa yang berbeda dan terdiri lebih dari 4300 binatang. Termasuk didalamnya satwa langka Indonesia maupun dunia yang terdiri dari Mamalia, Aves, Reptilia, Pisces. [2]

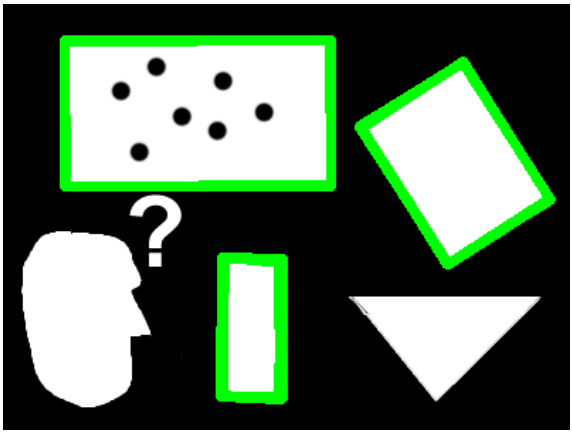
Gambar 2.1 Peta Wilayah Kebun Binatang Surabaya **[2]**

Mengunjungi KBS merupakan bagian pendidikan yang secara tidak langsung berguna untuk mengenal berbagai macam satwa yang ada supaya tertanam sejak dini perasaan mencintai seluruh alam dan isinya. Selain itu, KBS merupakan taman satwa yang artinya tempat atau wadah dengan fungsi utama konservasi ex-situ yang melakukan usaha perawatan dan penangkaran berbagai jenis satwa dalam rangka membentuk dan mengembangkan habitat baru sebagai sarana perlindungan dan pelestarian alam yang dimanfaatkan untuk pengembangan IPTEK serta untuk sarana rekreasi alam yang sehat.

Sasaran akhir taman satwa ini adalah memperluas pemahaman dan apresiasi masyarakat tentang fungsi taman satwa, meningkatkan upaya kesejahteraan satwa, menciptakan kaitan antara konservasi ex-situ dengan in-situ, membentuk jaringan global antar taman satwa. Program pendidikan dan penelitian serta melaksanakan pengembangan ilmu pengetahuan dan teknologi di Kebun Binatang seluas 15 hektar tersebut merupakan wahana keilmuan bagi masyarakat dan merupakan laboratorium hidup untuk lebih mencintai dan menghargai flora dan fauna sebagai kekayaan alam milik kita bersama.

2.3. OpenCV

OpenCV (Open Source Computer Vision) adalah suatu library yang berisi berbagai fungsi pemrograman untuk keperluan pengolahan citra pada mesin komputer secara *realtime*. [3] Pengolahan citra pada mesin komputer memungkinkan komputer dapat memproses data gambar yang diterima dan mengolahnya untuk mengenali suatu objek yang diinginkan. Beberapa contoh aplikasi dari OpenCV tersebut ialah tentang Face Recognition, Face Detection, Face/object Tracking, Road Tracking, dll.



Gambar 2.2 Contoh aplikasi OpenCV untuk *Shape Detection*

Jenis lisensi dari OpenCV adalah lisensi BSD dan digratiskan untuk semua tujuan, baik di bidang pendidikan maupun bidang komersial. OpenCV telah tersedia dalam berbagai bahasa pemrograman yaitu C++, C, Python, dan Java (android), serta dapat digunakan di berbagai jenis sistem operasi yaitu windows, linux, android, iOS, dan MacOS.

Beberapa fitur yang tersedia dalam library OpenCV adalah :

- Image Manipulation
  Pada fitur ini dapat digunakan fungsi untuk mengatur alokasi gambar, *copyingimage*, *setting*, serta *conversion*
- Image I/O
  Pada fitur ini memungkinkan pengguna untuk memilih input gambar yang akan diproses atau disimpan. Gambar

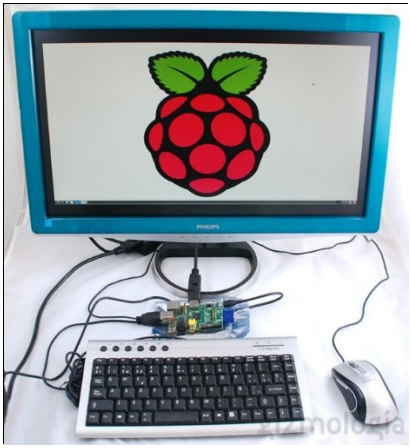
bisa didapatkan dari file foto, video, ataupun langsung melalui web-camera yang telah terkoneksi.

- Manipulasi Matriks dan Vektor

2.4. Raspberry-pi

Raspberry Pi adalah suatu single board computer seukuran kartu kredit yang dikembangkan dari Inggris oleh Raspberry Pi Foundation. [4] Layaknya sebuah komputer pada umumnya, Raspberry Pi juga menggunakan monitor, mouse dan keyboard serta perlengkapan komputer lainnya untuk dapat beroprasi.

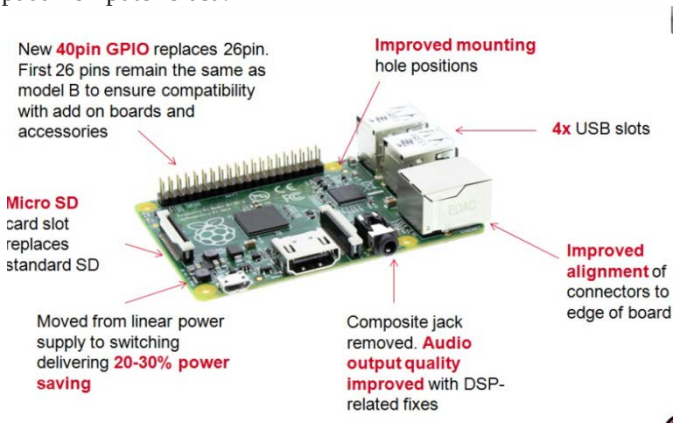
Raspberry Pi diciptakan dengan tujuan sebagai media pembelajaran dasar komputer. Dari segi harga, Raspberry Pi memiliki harga yang lebih ekonomis dibandingkan komputer pada umumnya sehingga Raspberry Pi ini sangat tepat apabila digunakan sebagai media pembelajaran.



Gambar 2.3 Penggunaan Raspberry Pi

Raspberry Pi dapat berjalan dengan menggunakan beberapa sistem operasi. Salah satu sistem operasi diantaranya adalah Raspbian. Raspbian adalah sistem operasi pada raspberry dengan basis linux yang menyerupai sistem operasi debian. Sistem operasi Raspbian dapat digunakan untuk banyak hal, mulai dari peggunaan dasar untuk

membuat dokumen, menjelajah internet, hingga memainkan game layaknya pada komputer biasa.



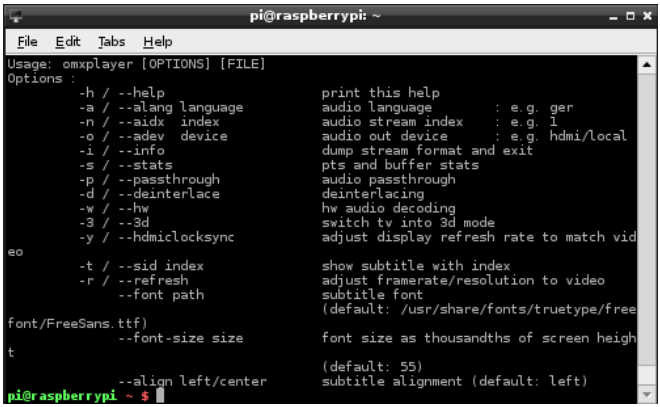
Gambar 2.4 Penjelasan Bagian Raspberry Pi

Namun apabila dibandingkan dengan computer biasa pada umumnya, dapat dikatakan bahwa spesifikasi hardware dari raspberry pi ini sangat kecil, sehingga tidak dapat digunakan untuk benar-benar menggantikan fungsi computer desktop pada umumnya. Namun daya listrik yang dibutuhkan pada Raspberry Pi sangat kecil apabila dibandingkan dengan daya listrik yang dibutuhkan oleh computer pada umumnya. Pada Raspberry-pi, daya listrik yang digunakan hanya bernilai 5V dengan arus kurang lebih 1.5A.

Produk dari Raspberry Pi mengalami perubahan versi dalam perkembangannya. Versi tersebut diantaranya yaitu Raspberry Pi B, Raspberry Pi B+, Raspberry Pi 2, Raspberry Pi A, dan lain sebagainya. Pada versi terbaru dari Raspberry Pi yaitu Raspberry Pi 2 telah memiliki spesifikasi prosesor dengan kecepatan 900MHz quadcore.

2.5. Omxplayer

OMXPlayer adalah aplikasi pemutar video dengan basis *command line* untuk Raspberry Pi. [5] Omxplayer berjalan pada Raspberry Pi dengan menggunakan sistem operasi raspbian. Omxplayer merupakan aplikasi yang bersifat *open-source*, dan sudah tersedia dalam instalasi awal dari sistem operasi raspbian.



Gambar 2.5 Tampilan menu Omxplayer

Omxplayer memiliki performa yang baik dalam menjalankan video di Raspberry Pi. Performa pemutaran media video oleh Omxplayer pada Raspberry dapat berjalan dengan baik karena aplikasi ini menggunakan GPU (*Graphic Processor Unit*) yang terdapat pada Raspberry Pi sehingga proses pemutaran video yang dilakukan pada Raspberry Pi tidak memberatkan kinerja dari CPU pada Raspberry Pi.

Omxplayer pada raspberry pi berjalan melalui terminal pada raspberry. Meski tidak memiliki GUI, namun aplikasi tersebut memiliki performa yang baik dan dapat digunakan untuk memutar media audio video dengan berbagai format, serta dapat memutar video dengan berbagai kualitas hingga kualitas HD.

2.6. Head-mounted Display

Head-mounted Display (HMD) merupakan perangkat *display* yang memproyeksikan informasi digital dari suatu data video atau suatu *wearable computer* langsung ke mata penggunanya. HMD dapat digunakan dalam berbagai aplikasi seperti pada bidang pembelajaran hingga pada bidang hiburan permainan. Setiap jenis dari HMD memiliki spesifikasi dan keunggulan yang berbeda-beda tergantung pada tujuan penggunaannya.[6]

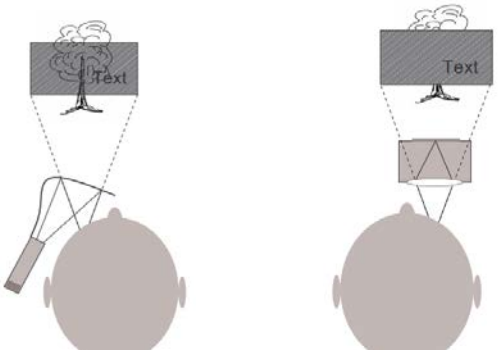
Gambar 2.6 Contoh Perangkat *Head-mounted Display*

HMD memiliki berbagai jenis untuk berbagai kebutuhan. Pada semua jenis HMD, memiliki karakteristik yaitu dapat menghasilkan gambar dari hasil data perangkat elektronik. Pada jenis HMD untuk video biasa, hanya disertai dengan layar video saja. Namun pada HMD yang diciptakan untuk Virtual Reality disertai dengan perangkat *head tracking* agar dapat mendeteksi arah gerakan kepala pengguna. Sedangkan pada HMD yang digunakan untuk *Augmented reality* memiliki kemampuan untuk merefleksikan gambar buatan dari data elektronik dan meneruskan cahaya dari dengan dunia nyata sehingga didapatkan hasil tampilan yang merupakan gabungan dari gambar buatan dan dunia nyata.[6]

Komponen utama dari HMD adalah satu atau dua layar tampilan, satu atau dua sistem optik untuk merefleksikan gambar, dan sebuah kerangka perangkat berupa helm. Jenis dari HMD dengan satu layar tampilan dan satu sistem optik disebut dengan *monocular* HMD. Pada jenis ini hanya digunakan satu gambar, sehingga tidak dapat dihasilkan sistem gambar stereo pada perangkat HMD tersebut. Sedangkan jenis HMD dengan dua layar tampilan dan dua sistem optik disebut dengan *binocular* HMD. Pada HMD jenis tersebut memiliki dua hasil tampilan yang sama namun sedikit berbeda antara satu dengan yang lain dengan efek 3-dimensi sehingga dapat menghasilkan gambar stereo.

Dari jenis sistem optik yang digunakan, HMD terbagi menjadi 2 jenis yaitu jenis *Totally Immersive* HMD (TI-HMD) dan jenis *See-Throught* HMD (ST-HMD). Pada jenis TI-HMD, pengguna tidak dapat melihat ke dunia nyata, sehingga seluruh pandangan pengguna akan

diarahkan pada gambar buatan dari data elektronik yang telah ditampilkan pada perangkat HMD tersebut. Sedangkan pada ST-HMD memiliki karakteristik sebaliknya, yaitu pengguna dapat melihat hasil gambar dari perangkat HMD dan tetap dapat melihat dunia nyata.

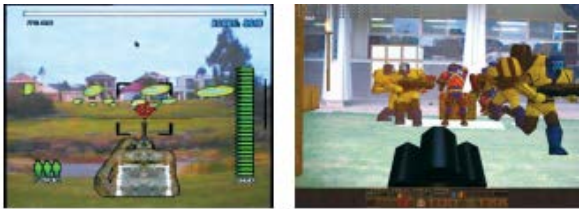


(a) (b)

Gambar 2.7 Jenis HMD (a) ST-HMD dan (b) TI-HMD

2.7. *Augmented reality*

Augmented reality (AR) memiliki konsep memasukkan data elektronik buatan ke dalam dunia nyata[7]. Teknologi *Augmented reality* dapat digunakan dalam kehidupan sehari-hari pada berbagai bidang seperti bidang kesehatan, bidang arsitektur, bidang pendidikan, hingga pada bidang industri hiburan.



(a) (b)

Gambar 2.8 Contoh aplikasi AR di bidang hiburan seperti permainan (a) ARQuake dan (b) Sky Invader

Banyak hal yang dapat dilakukan dengan teknologi *Augmented reality*. Mulai dari menampilkan informasi produk, menampilkan konsep desain tiga dimensi, hingga menampilkan navigasi arah.

(a) (b)

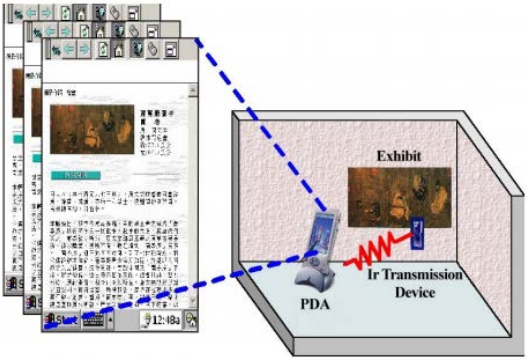
Gambar 2.9 *Augmented reality* (a) dengan menggunakan helm dan tas di lingkungan outdoor dan (b) hasil augmentasi gambar tampilan *Augmented reality***[7]**

Pada gambar di atas ditunjukkan contoh aplikasi dari penggunaan AR untuk menampilkan navigasi arah. Pada perangkat AR tersebut, tampilan navigasi arah akan dapat dilihat secara langsung oleh pengguna dan tampilan akan berubah sesuai dengan pergerakan kepala serta posisi dari pengguna. Sehingga selain dapat menampilkan tampilan navigasi secara langsung, perangkat HMD tersebut juga dapat menampilkan tampilan interaktif terhadap pergerakan penggunanya.

2.8. Pemandu Wisata Digital

Konsep pemandu wisata digital telah cukup banyak diciptakan untuk berbagai tempat pariwisata, seperti museum hingga kebun binatang. Berbagai konsep sistem pemandu wisata yang telah diciptakan tersebut menggunakan perangkat yang berbeda-beda, mulai dari hanya menggunakan *tape-machine*, cd-player, dan lain sebagainya. Salah satu sistem pemandu wisata digital yang telah memanfaatkan teknologi terbaru adalah sistem pemandu wisata untuk museum dengan menggunakan perangkat *Personal Digital Assistant* (PDA)[6].

Pada sistem tersebut, digunakan PDA sebagai perangkat utama yang dapat menampilkan informasi audio visual kepada penggunanya. Berikut merupakan gambar konsep dari sistem pemandu wisata tersebut.



Gambar 2.10 Konsep sistem pemandu wisata untuk museum

Perangkat tersebut bekerja dengan memanfaatkan transmitter infra-merah sebagai acuan untuk mendeteksi setiap objek di museum. Setiap kali perangkat PDA yang dipegang oleh pengguna mendeteksi sinyal infra-merah, maka perangkat PDA tersebut akan memutar informasi sesuai dengan nilai sinyal infra-merah yang telah dideteksi tersebut. Dengan demikian, maka perangkat PDA tersebut dapat memberikan informasi pada pengguna secara otomatis sesuai dengan objek yang dideteksi, sehingga pengguna tidak perlu direpotkan untuk mencari informasi yang sesuai dengan objek yang sedang dilihatnya.

Gambar 2.11 PDA sebagai perangkat sistem pemandu wisata

Pada contoh pembuatan sistem pemandu wisata lainnya, digunakan sistem pemandu wisata dengan memanfaatkan teknologi *geographic information system* (GIS). Penggunaan teknologi GIS pada dasarnya memanfaatkan *landscape mapping* untuk memetakan ruang dari objek wisata yang dituju, serta menggunakan GPS untuk menentukan posisi dari perangkat pengguna. Dengan menggunakan tenologi GIS tersebut, maka tidak diperlukan lagi penanda di setiap objek wisata.

BAB 3
PERANCANGAN SISTEM

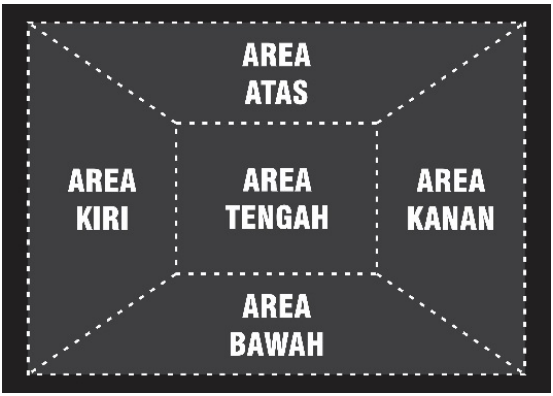
Pada bab ini akan dijelaskan mengenai perancangan sistem secara keseluruhan yang mencakup tentang konsep, perancangan, dan pembuatan sistem. Pembuatan sistem ditujukan untuk tempat wisata kebun binatang dengan lokasi studi kasus di Kebun Binatang Surabaya.

Gambar 3.1 Contoh Penempatan Marker

Dalam membedakan informasi yang akan ditampilkan secara otomatis di setiap kandang satwa, dibutuhkan suatu metode yang dapat membantu perangkat untuk membedakan kandang satwa tersebut. Salah satu metode yang dapat digunakan adalah dengan menggunakan gambar *marker* yang memiliki tanda berbeda dan dapat dipasang di setiap kandang satwa.

Untuk menghasilkan pemandu wisata digital yang interaktif, dibutuhkan tampilan yang interaktif pula. Untuk itu, maka digunakan adanya pilihan informasi yang interaktif, yang bisa dipilih dengan menggerakkan kamera ke arah tertentu terhadap gambar *marker*.

Area hasil gambar yang ditangkap oleh kamera akan dibagi menjadi lima bagian, yaitu area kanan, area kiri, area atas, area bawah, dan area tengah. Pembagian wilayah area tersebut dapat ditunjukkan pada gambar berikut.



Gambar 3.2 Pembagian area deteksi dari kamera

Kelima area tersebut akan memberikan informasi audio visual yang berbeda-beda kepada pengguna perangkat ini. Dengan mengarahkan perangkat pemandu wisata digital ke salah satu dari lima arah area tersebut pengguna dapat memilih informasi audio visual yang diinginkan.

Jenis informasi audio visual yang disajikan dalam perangkat ini berisi tentang berbagai informasi terkait Kebun Binatang Surabaya. Beberapa informasi yang dapat ditampilkan oleh perangkat pemandu wisata digital ini meliputi:

- Status satwa
- Penjelasan satwa
- Keunikan satwa
- Jenis satwa
- Peta (pemandu arah)
- Penjelasan panggung pertunjukan
- Jadwal pertunjukan

Salah satu contoh tampilan media audio visual yang terdapat pada perangkat ini adalah informasi mengenai salah satu satwa yang ada di Kebun Binatang Surabaya, yaitu satwa harimau dengan jenis Bengal. Pada media audio visual satwa tersebut memiliki tampilan awal berupa pilihan menu informasi yang diinginkan, yang ditunjukkan pada gambar berikut:

Gambar 3.3 Contoh tampilan awal informasi satwa

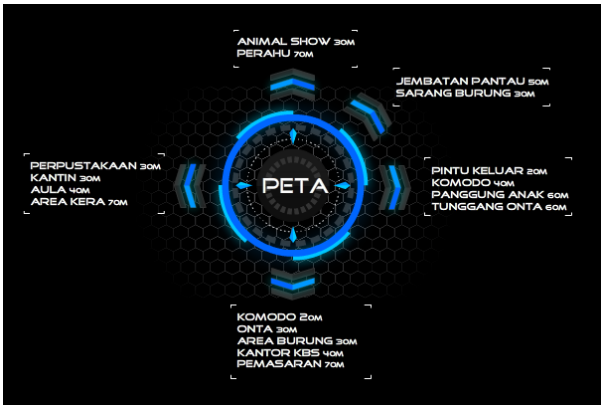
Pada tampilan audio visual yang berisi tentang informasi harimau tersebut, terbagi menjadi 4 bagian informasi yaitu:

- Informasi fakta unik satwa pada area kiri
- Informasi habitat satwa pada area atas
- Informasi deskripsi satwa pada area kanan
- Informasi status satwa pada area bawah

Sebagai contoh, apabila pengguna ingin mendapatkan informasi mengenai fakta unik tentang satwa harimau Bengal tersebut, maka pengguna hanya perlu sedikit menoleh sedikit ke arah kiri. Dengan demikian, maka perangkat pemandu wisata tersebut akan memunculkan tampilan yang berganti secara otomatis untuk menampilkan data video terkait dengan informasi fakta unik satwa harimau Bengal tersebut.

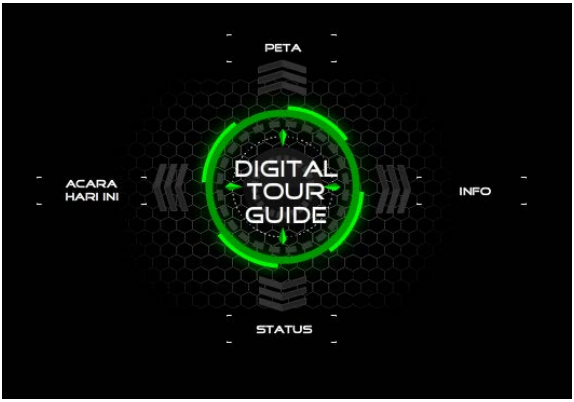

Gambar 3.4 Contoh video informasi satwa

Dan salah satu tampilan audio visual dari informasi penunjuk arah yang terdapat pada perangkat ini dapat dilihat pada gambar berikut



Gambar 3.5 Contoh tampilan peta penunjuk arah

Pada perangkat ini juga dapat disertakan informasi mengenai Kebun Binatang Surabaya itu sendiri. Seperti pada gambar berikut



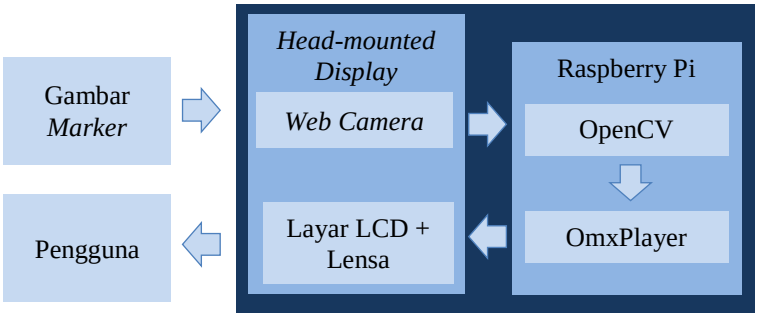
Gambar 3.6 Contoh tampilan menu informasi

Sehingga apabila terdapat gambar *marker* pada papan yang terdapat di area Kebun Binatang Surabaya, maka informasi tersebut

akan langsung dimunculkan pada perangkat pemandu wisata digital tersebut.

3.1. Diagram Blok Sistem

Diagram sistem secara keseluruhan dari perangkat pemandu wisata digital ini dapat dilihat pada gambar berikut:



Gambar 3.7 Diagram blok sistem keseluruhan

Raspberry Pi akan digunakan sebagai bagian utama dari system perangkat ini. Pada Raspberry Pi, akan dijalankan program OpenCV untuk mendeteksi gambar *marker* yang didapat dari kamera webcam yang telah terhubung dengan Raspberry Pi tersebut.

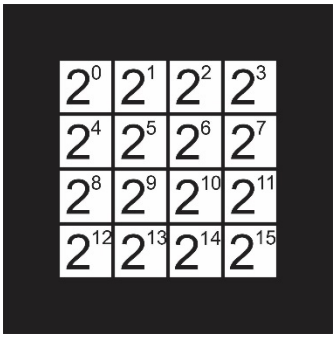
Setelah OpenCV mendeteksi adanya gambar *marker* dan mendapatkan koordinat posisinya terhadap kamera, maka data tersebut akan dikirimkan keluar dari program OpenCV kepada program pemutar media digital OMX Player sebagai referensi koordinat untuk memutar file audio visual yang sesuai.

3.2. Desain Gambar *Marker*

Gambar *marker* yang ingin dideteksi melalui pengolahan citra OpenCV pada Raspberry Pi tersebut didesain dengan bentuk dasar persegi dimana didalamnya dibagi menjadi 4x4 bagian yang dapat merepresentasikan data sebesar 16 bit. Dengan desain tersebut, maka dapat digunakan marker dengan banyak nilai yang berbeda hingga 6500 marker.

Gambar 3.8 Desain gambar marker

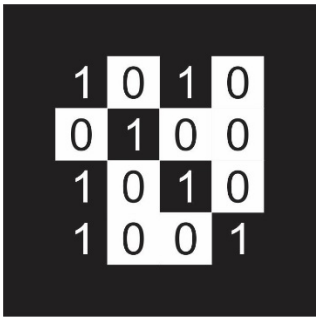
Bentuk gambar *marker* tersebut memiliki 16 persegi di dalam 1 bingkai persegi besar. Nilai dari 16 persegi yang berada di dalam akan mewakili masing-masing bit sesuai dengan ilustrasi berikut :



Gambar 3.9 Ilustrasi nilai bit pada gambar *marker*

Apabila warna salah satu persegi pada *marker* berwarna hitam, maka akan merepresentasikan nilai bit pada bagian tersebut dengan nilai 1. Sedangkan apabila warna persegi berwarna putih, maka akan merepresentasikan nilai bit pada bagian tersebut dengan nilai 0.

Salah satu contoh dari nilai bit yang terkandung di dalam gambar *marker* dapat kita lihat pada gambar berikut.



Gambar 3.10 Ilustrasi contoh nilai gambar marker

Pada contoh ilustrasi gambar *marker* di atas memiliki nilai data bit sebagai berikut:

Tabel 3.1 Tabel ilustrasi nilai bit pada gambar marker

		warna	nilai
Bit	0	hitam	1
	1	putih	0
	2	hitam	1
	3	putih	0
	4	putih	0
	5	hitam	1
	6	putih	0
	7	putih	0
	8	hitam	1
	9	putih	0
	10	hitam	1
	11	putih	0
	12	hitam	1
	13	putih	0
	14	putih	0
	15	hitam	1

Sehingga nilai biner yang terkandung dalam gambar tersebut adalah 1010010010101001. Dan jika bilangan biner tersebut dikonversi ke dalam bentuk bilangan decimal, maka didapatkan nilai 42153.

Hasil desain marker tersebut kemudian akan dicetak dalam ukuran 20cm x 20cm dan diletakkan pada papan penjelasan yang terdapat di Kebun Binatang Surabaya.

Gambar 3.11 Penempatan Marker pada papan penjelasan di Kebun Binatang Surabaya

3.3. Web Camera

Pada pembuatan tugas akhir ini digunakan *web camera* logitech dengan type C170. W*eb camera* jenis ini telah memiliki fitur *autofocus* yang dapat menyesuaikan titik fokus pada gambar marker yang akan dideteksi.

Gambar 3.12 Web Camera Logitech C170 **[7]**

Berikut merupakan fitur keseluruhan dari Logitech webcamera C170:

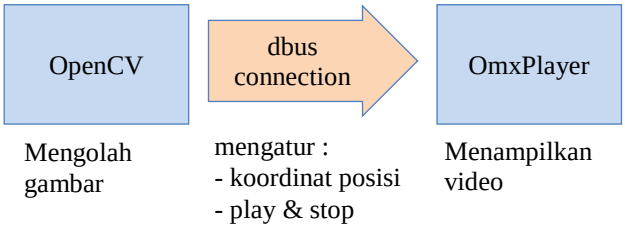
- Resolusi 5 megapiksel
- Pengaturan *plug and play*
- Mampu merekam video dengan kualitas XVGA
- Hasil rekam video dengan ukuran 1024x768
- Memiliki mikrofon bawaan dengan pengurangan noise

3.4. Raspberry Pi B+

Pada pembuatan tugas akhir ini, Raspberry Pi digunakan sebagai unit pemroses utama untuk menjalankan sistem keseluruhan. Jenis Raspberry Pi yang digunakan pada tugas akhir ini adalah Raspberry Pi B+. Raspberry Pi B+ memiliki spesifikasi sebagai berikut :

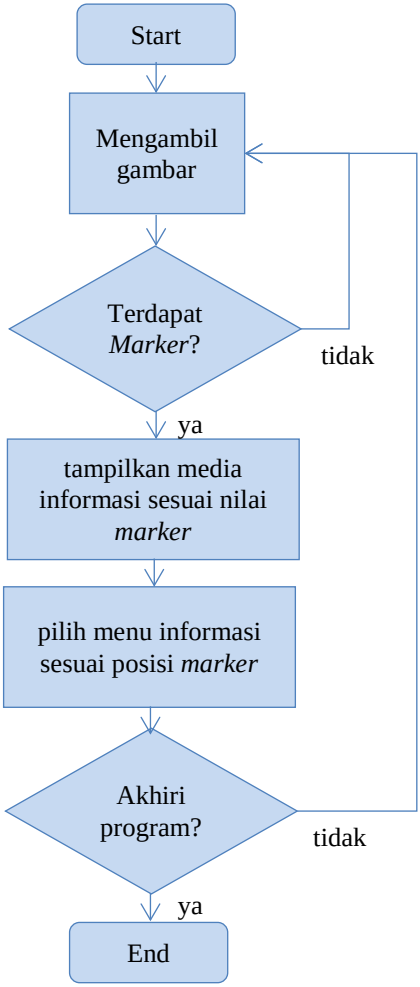
- Memory Micro SD Card slot 1x
- RAM Memori 512 RAM
- Slot USB 2.0 4x
- Slot Ethernet 1x
- GPIO
- Clock 700 MHz

Raspberry Pi digunakan untuk memproses data pengolahan citra dengan OpenCV dan menampilkan video informasi dengan OmxPlayer.



Gambar 3.13 Blok diagram proses pada Raspberry Pi

Karena proses pengolahan data pada Raspberry Pi terdiri dari dua program, maka dibutuhkan suatu penghubung yang dapat mengirimkan data antar program yang dijalankan pada Raspberry Pi tersebut. Untuk menghubungkan komunikasi antara dua program tersebut di dalam Raspberry Pi, digunakan metode dbus connection.



Gambar 3.14 Flowchart proses data pada Raspberry Pi

3.4.1. OpenCV

3.4.1.1. Mendeteksi Bentuk Persegi

Untuk mendeteksi gambar marker yang telah diterima dari kamera usb, pertama-tama diperlukan pengkondisian gambar terlebih dahulu.

Gambar 3.15 Hasil gambar yang ditangkap oleh kamera

Pengkondisian gambar tersebut dilakukan dengan beberapa tahapan. Tahapan yang pertama yaitu tahap pengkonversian gambar menjadi grayscale. Pada tahap ini, warna pada gambar akan dihilangkan, dan diubah menjadi gambar grayscale.

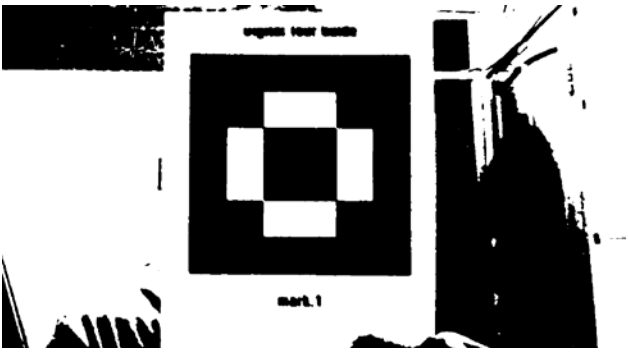
Gambar 3.16 Hasil konversi gambar menjadi grayscale

Pengubahan gambar menjadi *grayscale* pada tugas akhir ini dilakukan dengan menggunakan fungsi yang telah disediakan oleh OpenCV dengan perintah berikut:

```
cvCvtColor(img, gray, CV_RGB2GRAY);
```

Dengan variabel img berupa variable IplImage gambar sumber yang berasal dari kamera, dan variable gray berupa variable IplImage yang berisi hasil konversi *grayscale* dari gambar sumber. Pada perintah tersebut juga ditambahkan perintah CV_RGB2GRAY sebagai penanda untuk mengkonversikan gambar dari jenis RGB menjadi grayscale.

Pada tahap berikutnya, dilakukan *thresholding* pada gambar yang telah menjadi grayscale tersebut. Pada tahap thresholding ini, akan dihasilkan gambar biner dengan warna hitam dan putih. Metode thresholding memiliki banyak jenis yang dapat digunakan. Salah satu dari jenis metode thresholding yang digunakan pada penulisan tugas akhir ini adalah metode OTSU thresholding. Metode OTSU thresholding merupakan salah satu metode thresholding dengan kelebihan mampu melakukan thresholding dengan lebih baik pada keadaan kondisi cahaya yang tidak stabil. Pada metode ini, apabila cahaya gambar yang diterima lebih terang atau lebih gelap, metode OTSU thresholding dapat menentukan batas warna thresholding secara otomatis berdasarkan histogram gambar grayscale yang ingin diubah.



Gambar 3.17 Hasil konversi gambar menjadi binary image

Pada pembuatan tugas akhir ini, digunakan perintah OTSU thresholding dengan menggunakan fungsi yang telah ada pada library OpenCV, yaitu dengan menggunakan perintah berikut:

```
cvThreshold(gray, thres, 1, 255, CV_THRESH_OTSU);
```

Variable gray pada perintah tersebut berupa variable dengan jenis IplImage yang berisi gambar hasil *grayscale* pada tahap

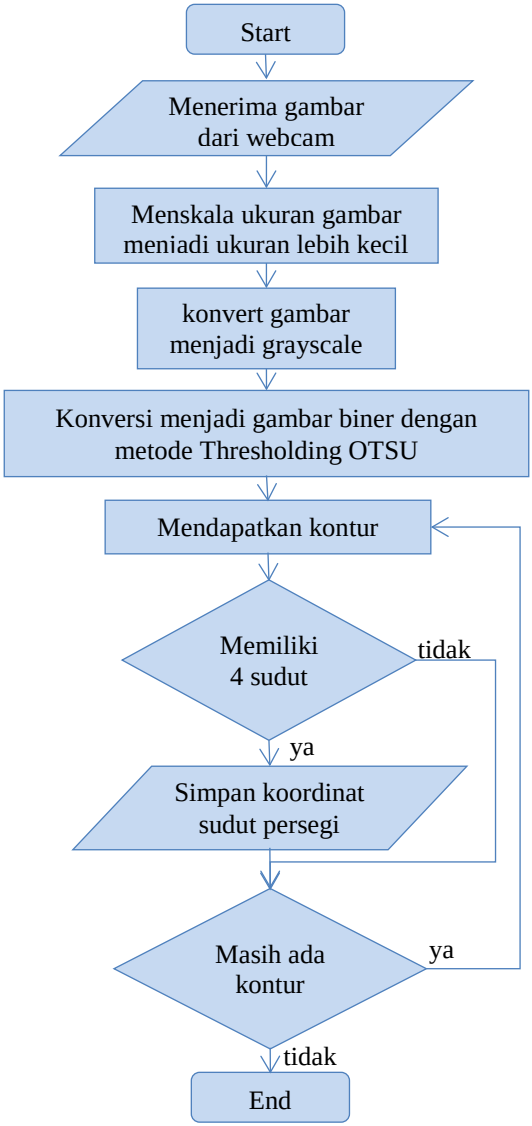
sebelumnya. Sedangkan variable thres pada perintah tersebut berupa variable dengan jenis IplImage yang berisi gambar hasil pengkonversian grayscale menjadi gambar biner. Nilai parameter setelah variable threshold merupakan batas nilai yang digunakan sebagai acuan pengkonversian gambar menjadi gambar biner. Namun karena metode OTSU thresholding secara otomatis dapat menentukan nilai batas thresholding dari analisa histogram.

Setelah melalui proses pengkondisian, maka gambar tersebut akan dideteksi melalui metode *contour*. Pada tahapan ini, akan dideteksi bentuk-bentuk objek pada gambar berdasarkan tepi perbedaan warna dari hasil thresholding sebelumnya.

Setelah dilakukan contour, maka tahap berikutnya adalah penentuan bentuk persegi. Bentuk persegi merupakan bentuk yang memiliki 4 sudut, dimana masing-masing sudutnya memiliki nilai sudut sebesar 90 derajat. Oleh karena itu, dalam penentuan bentuk persegi tersebut, dilakukan dengan memeriksa jumlah sudut pada tiap bentuk objek yang ditemukan pada tahap contour.

Gambar 3.18 Hasil deteksi bentuk persegi pada gambar

Setiap hasil objek persegi yang telah dideteksi kemudian dianalisa lebih lanjut pada tahap berikutnya untuk mengetahui apakah objek persegi tersebut merupakan gambar marker atau bukan.



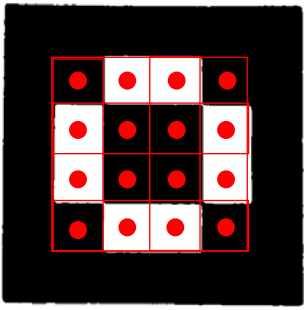
Gambar 3.19 Flowchart pendeteksian betuk persegi OpenCV

3.4.1.2. Mendeteksi Gambar Marker

Untuk mengetahui apakah objek persegi yang telah dideteksi merupakan suatu marker dengan nilai yang ada pada database atau bukan, maka diperlukan perintah untuk menganalisa objek persegi tersebut. Perintah untuk menganalisa apakah objek persegi tersebut berupa marker atau tidak diawali dengan mengambil hanya bagian objek persegi tersebut dan mentransformasikan objek tersebut menjadi persegi dengan resolusi 180x180.

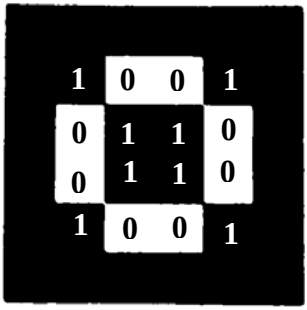
Gambar 3.20 pengambilan gambar objek persegi dan transformasi objek persegi

Setelah itu, objek persegi tersebut dibagi menjadi 6x6 bagian. Dengan demikian, didapatkan 16 titik bagian 4x4 pada sisi dalam yang berisi nilai marker yang ingin dideteksi.



Gambar 3.21 pembagian 16 titik pada marker

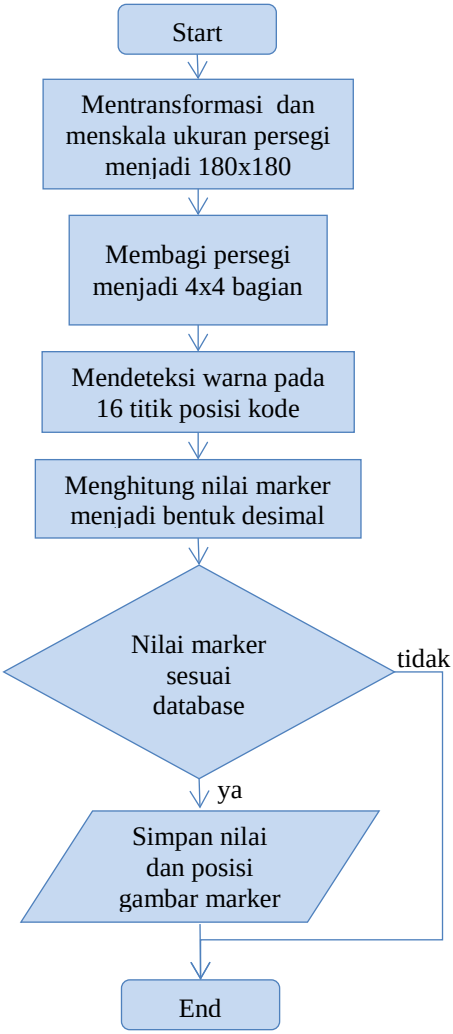
Tahap berikutnya adalah mendeteksi warna pixel pada setiap bagian tengah dari bagian persegi kecil 4x4 tersebut. Hasil pendeteksian warna tersebut akan menjadi suatu bilangan biner dengan ketentuan jika berwarna hitam merepresentasikan nilai 1 dan jika berwarna putih merepresentasikan nilai 0. Dengan mengurutkan nilai bit dari 16 titik tersebut, maka didapatkan hasil bilangan biner 16 bit yang merupakan nilai dari marker tersebut.



Gambar 3.22 Representasi nilai marker

Hasil dari bilangan biner tersebut kemudian dikonversikan menjadi bilangan integer dan dibandingan dengan nilai marker yang telah tersimpan pada database. Apabila nilai marker tersebut sesuai dengan nilai yang terdapat pada database, maka nilai marker yang telah dideteksi tersebut beserta posisi marker tersebut terhadap resolusi layar akan disimpan dan digunakan sebagai acuan untuk mengarahkan tampilan video OmxPlayer pada tahap berikutnya.

Sedangkan apabila tidak terdapat marker yang sesuai dengan nilai yang tersimpan pada database, maka OpenCV akan memerintahkan OMX Player untuk tidak memutar media apapun, sehingga ketika pengguna sedang tidak mengarahkan perangkat pemandu wisata pada marker, maka tidak aka nada video yang ditampilkan pada pengguna.



Gambar 3.23 flowchart proses deteksi gambar marker

3.4.1.3. Mengirimkan perintah ke OmxPlayer

Hasil pendeteksian marker pada OpenCV menghasilkan dua nilai variable yang digunakan sebagai acuan untuk mengatur tampilan video pada OmxPlayer. Kedua variable tersebut adalah variable koordinat posisi marker terhadap resolusi layar LCD, dan variable jenis video yang ingin ditampilkan berdasarkan nilai marker yang telah dideteksi.

3.4.2. OmxPlayer

OmxPlayer pada Raspberry Pi dapat dikontrol dengan menggunakan beberapa cara. Beberapa cara untuk mengontrol Omxplayer tersebut diantaranya adalah dengan menggunakan tombol pada *keyboard* dan menggunakan perintah melalui dbus connection. Untuk dapat mengontrol tampilan video pada Omxplayer secara otomatis melalui hasil deteksi OpenCV, maka digunakan kontrol dengan memanfaatkan dbus connection.

3.4.3. Dbus Connection

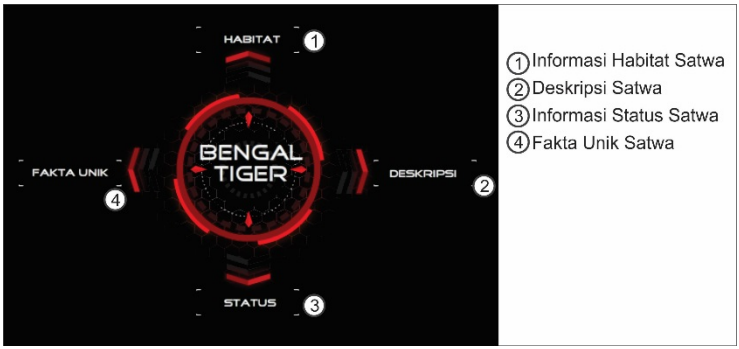
Dbus Connection adalah suatu proses komunikasi antar program dengan cara membagikan informasi tertentu pada mesin yang sama. Dengan adanya Dbus Connection tersebut, maka hasil pengolahan data pada OpenCV dapat digunakan untuk mengontrol tampilan yang diproses pada OmxPlayer.

Perintah dbus pada Omxplayer yang digunakan oleh OpenCV mencakup tentang :

- Set video pos: merupakan perintah yang digunakan untuk mengatur posisi waktu penampilan video sesuai dengan yang diinginkan.
- Set video position: merupakan perintah yang digunakan untuk mengatur posisi koordinat penampilan video pada layar LCD.
- Hide/unhide video: merupakan perintah yang digunakan untuk menentukan apakah video ditampilkan atau tidak

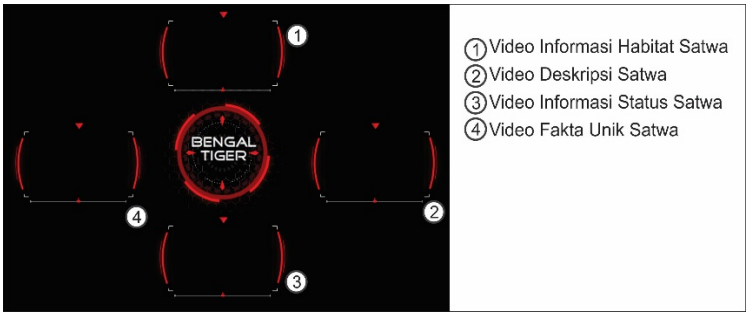
3.4.4. Desain File Video

Tampilan pada OmxPlayer memiliki beberapa bagian, dengan tampilan awal dengan beberapa menu pilihan informasi



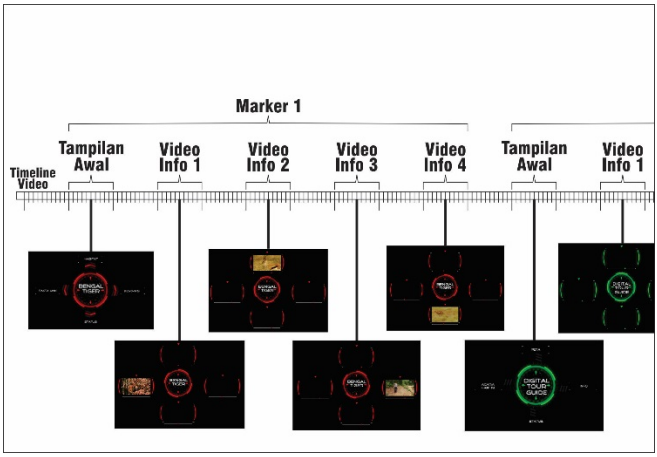
Gambar 3.24 Desain tampilan awal informasi satwa

Masing-masing pada menu pilihan informasi tersebut berisikan data video yang akan diputar apabila menu tersebut dipilih. Video informasi tersebut disajikan melalui tampilan audio visual, dengan penjelasan yang lengkap.



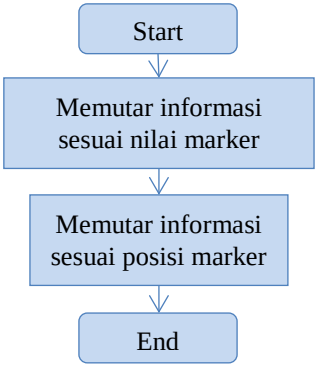
Gambar 3.25 Desain tampian informasi video informasi satwa

Dari beberapa bagian video tersebut, digabungkan menjadi satu file video dengan penentuan waktu dan jeda waktu yang diatur, sehingga pemilihan bagian video yang ingin ditampilkan dapat ditentukan dengan menentukan waktu penampilan pada video tersebut. Pembagian penentuan waktu penampilan video pada file video dapat dilihat pada gambar berikut:



Gambar 3.26 Timeline video informasi satwa

Sebagai contoh, apabila hasil dari pembacaan gambar *marker* oleh OpenCV menunjukkan gambar *marker* nomor 1 dan mengharuskan OmxPlayer memutar video informasi terkait habitat satwa, maka OmxPlayer hanya perlu memutar video pada menit ke 3 hingga menit ke 4 pada file video yang telah disimpan.



Gambar 3.27 Flowchart proses Omxplayer

3.5. Head-mounted Display

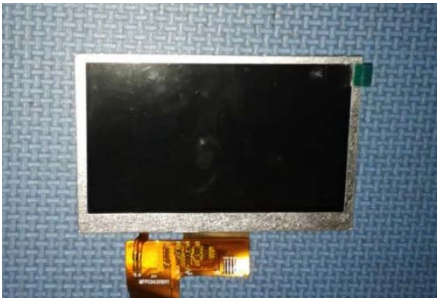
Untuk membuat perangkat yang sesuai dengan kondisi Kebun Binatang Surabaya, dilakukan analisa kondisi dari lokasi Kebun Binatang Surabaya tersebut dengan hasil berikut :

- Area Kebun Binatang Surabaya sangat luas hingga 15 hektar
- Jumlah hewan sangat banyak hingga 300 jenis satwa dan lebih dari 4000 ekor satwa.
- Hampir seluruh area dari Kebun Binatang Surabaya berada di area terbuka

Untuk membuat perangkat Head-mounted Display yang dapat dipakai untuk *augmented reality* maka digunakan prinsip lensa semi transparan yang dapat memantulkan sebagaian cahaya dan dapat meneruskan sebagian cahaya. Dengan menggunakan prinsip sederhana tersebut serta dengan kelengkungan lensa konveks-konkav yang menghasilkan titik fokus pantulan yang tepat, maka bayangan hasil pantulan dari layar LCD akan dapat diterima oleh mata pengguna tanpa menutupi pandangan pengguna secara total, sehinngga memungkinkan untuk dapat diciptakan perangkat *augmented reality* hardware tersebut.

3.5.1. Layar LCD

Layar yang digunakan pada perangkat ini adalah layar LCD dengan ukuran 4 inch dengan resolusi sebesar 640x480. Layar tersebut bekerja pada tegangan sumber sebesar 12 volt DC.



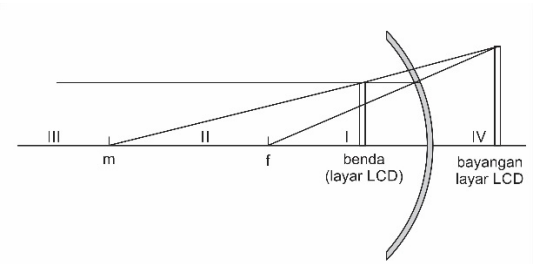
Gambar 3.28 Mini LCD 4 inch

Layar LCD tersebut dapat dihubungkan ke Raspberry Pi melalui jack audio video RCA pada Raspberry Pi, sehingga dibutuhkan kabel converter dari RCA menjadi jack 3.5mm audio video.

Setelah semua kabel dari layar LCD telah selesai dipasang, maka tahap terakhir adalah pengecekan hasil akhir dari hardware tersebut, yaitu dengan memeriksa hasil bayangan.

3.5.2. Lensa

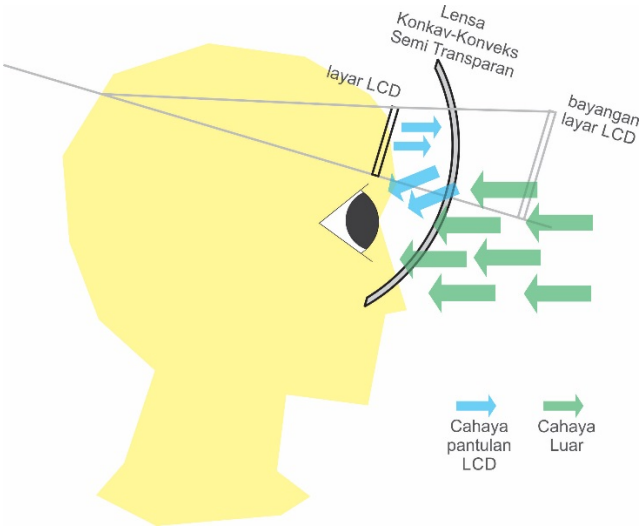
Lensa yang dibutuhkan pada perangkat HMD ini didesain dengan menggunakan prinsip penghitungan bayangan cermin cekung. Besarnya kelengkungan lensa serta jarak lensa dari layar LCD dapat dilihat pada ilustrasi berikut :



Gambar 3.29 penjelasan pemantulan cahaya pada cermin cekung

Dengan peletakan lensa konveks-konkav semi transparan di depan mata pengguna, maka cahaya yang akan masuk dan dilihat oleh pengguna akan bersumber dari dua hal. Yang pertama yaitu cahaya dari luar, yaitu cahaya dari pantulan matahari dan benda lainnya yang masuk ke mata selayaknya melihat seperti biasa. Dan yang kedua adalah cahaya gambar dari pantulan layar LCD yang terdapat pada perangkat *augmented reality* tersebut.

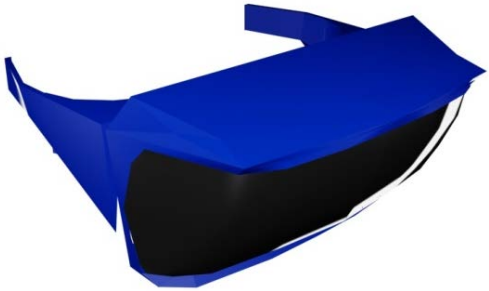
Dengan adanya bentuk lensa berjenis konveks-konkav, maka hasil bayangan layar LCD yang dipantulkan oleh lensa tersebut menjadi bersifat semu, diperbesar, dan berada pada ruang 4. Sehingga hasil akhir yang dilihat oleh pengguna dapat dikatakan merupakan penggabungan dari dua cahaya yang diterimanya.



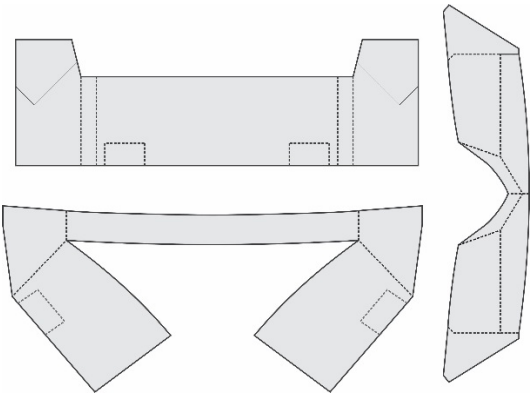
Gambar 3.30 konsep posisi peletakan lensa konveks-konkav

3.5.3. Casing Head-mounted Display

Untuk membuat HMD agar sesuai dengan jarak dan ukuran yang telah diuraikan sebelumnya, maka dibutuhkan pengukuran HMD yang harus disesuaikan dengan pengguna. Dalam penulisan buku tugas akhir ini, ukuran pengguna yang digunakan adalah penulis buku ini sendiri.

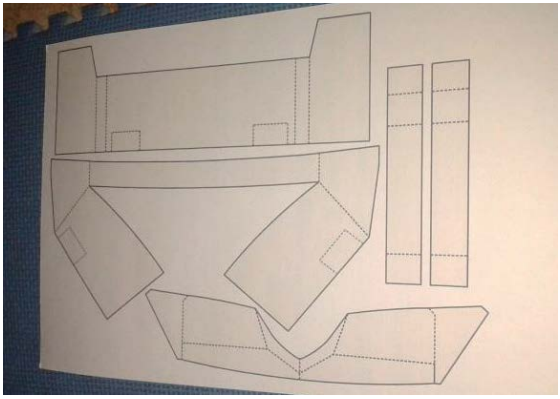


Gambar 3.31 Konsep Desain Perangkat HMD



Gambar 3.32 Desain papercraft perangkat HMD

Proses pembuatan HMD dilakukan dengan metode papercraft, yaitu diawali dengan design menggunakan kertas. Ukuran dari desain menggunakan metode papercraft ini tentunya menyesuaikan ukuran yang telah ditentukan pada penghitungan lensa sebelumnya.



Gambar 3.33 hasil cetak desain papercraft HMD

Hasil kertas yang telah berisi desain dari perangkat HMD tersebut kemudian dirangkai agar menjadi bentuk dasar dari perangkat HMD yang diinginkan. Berikut merupakan hasil dari bentuk dasar perangkat HMD dari bahan kertas tersebut.

Gambar 3.34 hasil pembuatan desain papercraft HMD

Setelah desain dengan menggunakan kertas tersebut dirasa telah sesuai, maka proses pembuatan HMD tersebut dilanjutkan dengan melapisi hasil desain papercraft tersebut dengan menggunakan bahan resin dan fiber.

Gambar 3.35 hasil pelapisan HMD dengan resin

Proses ini memerlukan waktu pengeringan hingga 2 jam. Dan kemudian untuk mendapatkan hasil yang maksimal, maka proses pelapisan dengan resin tersebut dilakukan secara berulang hingga 3 kali pelapisan.

Setelah lapisan resin telah benar-benar kering, tahap selanjutnya adalah tahap pengecatan agar perangkat HMD tersebut terlihat lebih rapi dan menarik.

Gambar 3.36 tampak depan hasil pembuatan HMD

Gambar 3.37 tampak belakang hasil pembuatan HMD

Penempatan kamera pada perangkat HMD tersebut diletakkan pada posisi atas dari bagian HMD. Untuk perangkat layar LCD, diletakkan sesuai dengan posisi yang telah ditentukan berdasarkan jarak dari lensa konveks-konkav pada HMD. Sedangkan posisi dari perangkat

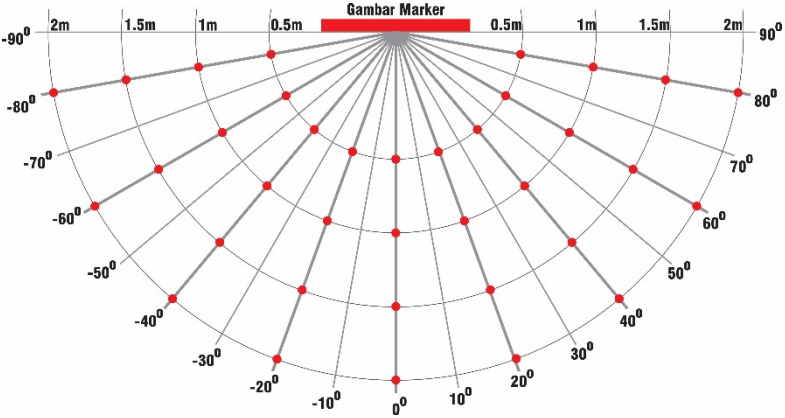
Raspberry Pi dan baterai diletakkan secara terpisah dari perangkat HMD, yaitu diletakkan pada pinggang pengguna sehingga dapat mengurangi beban yang terdapat pada perangkat HMD. Kemudian dari hasil perangkat HMD tersebut dipasangkan dengan pengait kepala dan komponen lainnya

BAB 4
PENGUJIAN DAN ANALISIS

Pada tugas akhir ini, dilakukan pengujian dengan menggunakan 3 variabel yang diubah, yaitu jarak, sudut, dan cahaya. Serta dilakukan pengujian untuk mengukur performa kerja perangkat dan hasil perbandingan perangkat dengan sistem pemandu wisata digital lain yang sudah ada.

4.1. Pengujian Jarak dan Sudut

Pada pengujian dengan variable jarak, ditentukan posisi pengujian jarak dengan interval 0.5 meter, 1 meter, 1.5 meter, dan 2 meter dari posisi gambar marker. Sedangkan pada pengujian dengan variable sudut, ditentukan posisi pengujian sudut dengan interval -80, -60, -40, -20, 0, 20, 40, 60, dan 80 derajat dari posisi gambar marker. Sehingga dari kedua variable tersebut didapatkan posisi pengujian. Posisi pengujian digambarkan sebagai titik merah pada gambar berikut ini:



Gambar 4.1 Posisi pengujian variabel jarak dan sudut

Pengujian dilakukan di Kebun Binatang Surabaya, dengan meletakkan marker pada papan penjelasan satwa yang ada. Pengujian

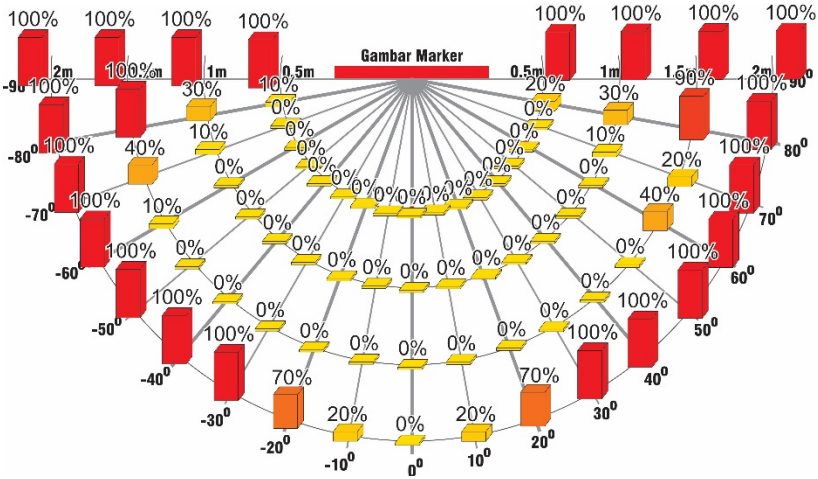
dilakukan pada siang hari, dengan intensitas cahaya matahari sekitar 2800 lux.

Gambar 4.2 Posisi peletakan gambar marker pada papan penjelasan satwa di Kebun Binatang Surabaya

Gambar 4.3 Proses pengujian di Kebun Binatang Surabaya

Pada pengujian tersebut, dilakukan dengan memposisikan kamera menghadap gambar marker. Pada posisi tersebut, dilakukan pendeteksian 10 *frame* gambar yang didapatkan dari kamera secara berurutan. Dari 10 *frame* gambar yang telah dideteksi tersebut didapatkan jumlah *frame* gambar yang berhasil dideteksi sebagai gambar marker oleh kamera, dan jumlah *frame* gambar yang tidak berhasil dideteksi.

Dari hasil pengujian dengan posisi tersebut, didapatkan jumlah *frame* gambar yang berhasil dideteksi dari 10 *frame* gambar yang diuji, yang dapat dilihat pada table berikut:



Gambar 4.4 Grafik error hasil pengujian jarak dan sudut

Dari pengujian jarak dan sudut tersebut, didapatkan rata-rata jarak error terbesar pada jarak 2 meter dan sudut error terbesar pada sudut +- 90 derajat.

4.2. Pengujian Performa Perangkat

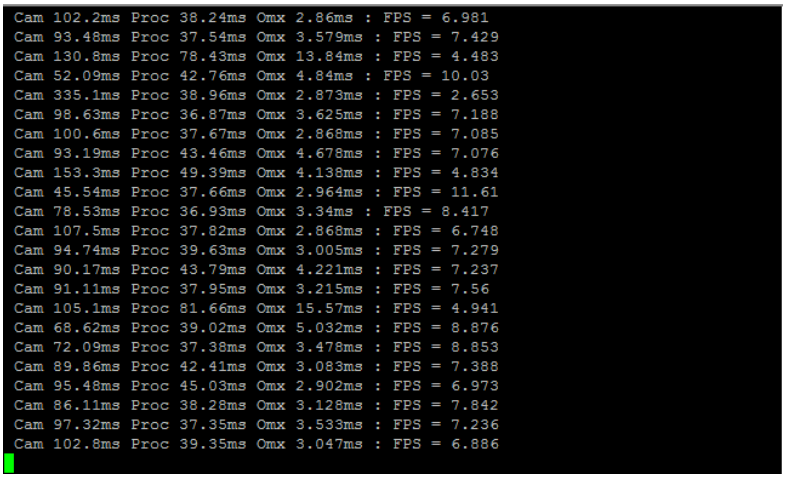
Pengujian performa dilakukan dengan memantau kecepatan proses pada Raspberry Pi yang ditampilkan pada terminal. Pengujian performa terbagi menjadi beberapa bagian, yaitu:

- Pengujian performa terhadap kecepatan perangkat secara keseluruhan
- Pengujian performa terhadap kecepatan pengambilan gambar oleh kamera
- Pengujian performa terhadap kecepatan pengolahan citra
- Pengujian performa terhadap kecepatan pengaturan OMXPlayer

Pengujian dilakukan dengan menghitung jumlah frame gambar yang dapat diolah setiap detik oleh sistem. Pengujian dilakukan dalam dua kondisi, yaitu kondisi ketika perangkat dalam keadaan tidak mendeteksi gambar marker dan kondisi ketika perangkat sedang

mendeteksi gambar marker. Pengujian dengan dua kondisi ini bertujuan untuk mengetahui pengaruh dari masing-masing bagian proses dari perangkat yang menentukan kecepatan performa perangkat terhadap pendeteksian gambar.

Pada kondisi ketika perangkat tidak sedang mendeteksi gambar marker, didapatkan hasil yang ditunjukkan pada gambar berikut:

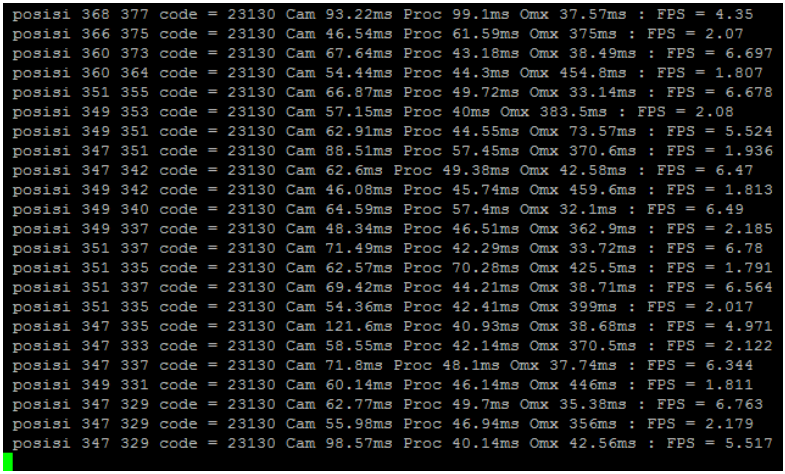


Gambar 4.5 Hasil pengujian perangkat ketika tidak mendeteksi gambar *marker*

Pada hasil pengujian perangkat ketika dalam kondisi tidak sedang mendeteksi gambar *marker*, didapatkan rata-rata waktu pengolahan total 151.5055 ms atau kecepatan sebesar 7.1884 fps dengan perincian waktu pengelolaan sebagai berikut:

- Rata-rata waktu untuk pengambilan gambar oleh *web-camera* sebesar 103.6678 ms
- Rata-rata waktu untuk pendeteksian gambar marker sebesar 43.37304 ms
- Rata-rata waktu untuk pengaturan Omxplayer sebesar 4.464652 ms

Pada pengujian kedua, yaitu dalam kondisi ketika perangkat sedang mendeteksi gambar marker, didapatkan hasil yang ditunjukkan pada gambar berikut:



Gambar 4.6 Hasil pengujian perangkat ketika mendeteksi gambar *marker*

Pada hasil pengujian perangkat ketika dalam kondisi tidak sedang mendeteksi gambar *marker*, didapatkan rata-rata waktu pengolahan total 329.8252 ms atau kecepatan sebesar 4.1282 fps dengan perincian waktu pengelolaan sebagai berikut:

- Rata-rata waktu untuk pengambilan gambar oleh *web-camera* sebesar 67.223 ms
- Rata-rata waktu untuk pendeteksian gambar marker sebesar 50.095 ms
- Rata-rata waktu untuk pengaturan Omxplayer sebesar 212.506 ms

4.3. Pengujian Cahaya

Untuk pengujian dengan pengaruh cahaya, dilakukan pengujian dengan mengubah-ubah kondisi cahaya saat pengujian. Besarnya intensitas cahaya pada saat pengujian diukur dengan menggunakan light meter. Pengujian ini bertujuan untuk mengetahui kondisi intensitas cahaya optimal yang diperlukan agar perangkat dapat bekerja dengan baik.

Pada pengujian cahaya tersebut, dilakukan dengan dua kondisi. Kondisi pertama dilakukan di luar ruangan pada siang hari untuk mendapatkan intensitas cahaya yang besar, yaitu mulai 500 lux hingga bernilai 3000 lux.

Gambar 4.7 Pengujian pengaruh kondisi cahaya di luar ruangan

Pada pengujian kondisi pertama didapatkan hasil yang dapat dilihat pada table berikut:

Tabel 4.1 Hasil pengujian pengaruh intensitas cahaya di luar ruangan

Intensitas Cahaya (lux)	Hasil Deteksi Gambar Marker	
	Terdeteksi	Tidak Terdeteksi
3000	√	
2000	√	
1000	√	
500	√	

Sedangkan pada kondisi kedua, dilakukan di dalam ruangan untuk memperoleh intensitas cahaya yang kecil, yaitu mulai 3 lux hingga bernilai 100 lux. Pengujian dilakukan dengan menempatkan perangkat pada posisi jarak dan sudut yang tetap, yaitu sudut 0 derajat menghadap ke gambar marker dan jarak 50 cm dari gambar marker.

Gambar 4.8*Pengujian pengaruh kondisi cahaya di dalam ruangan*

Hasil dari pengujian dengan cahaya dengan intensitas rendah di dalam ruangan tersebut dapat dilihat pada tabel berikut:

Tabel 4.2 Hasil pengujian intensitas cahaya di dalam ruangan

Intensitas Cahaya (lux)	Hasil Deteksi Gambar Marker	
	Terdeteksi	Tidak Terdeteksi
100	√	
50	√	
20	√	
10	√	
5		√
3		√

Dari hasil pengujian pengaruh intensitas cahaya pada tabel di atas, tampak bahwa perangkat pemandu wisata tersebut hanya dapat bekerja pada kondisi intensitas cahaya lebih minimal 10 lux. Dan perangkat tetap mampu bekerja secara optimal hingga pada intensitasa cahaya 3000 lux yang merupakan intensitas cahaya di luar ruangan.

4.4. Perbandingan dengan sistem pemandu wisata lain

Pada analisa berikut, ditunjukkan perbandingan hasil perangkat pembuatan sistem pemandu wisata digital ini dibandingkan dengan sistem pemandu wisata digital lainnya. Dan didapatkan hasil sebagai berikut:

Tabel 4.3 Analisa perbandingan dengan sistem pemandu wisata lain

	Pemandu Wisata	*Tape Machine*	*CD Player*	PDA	HMD
Waktu persiapan awal	tinggi	rendah	rendah	rendah	rendah
Biaya untuk perubahan informasi	tinggi	sedang	sedang	rendah	rendah
Biaya penggunaan jangka panjang	tinggi	rendah	rendah	rendah	rendah
Kesesuaian konten	tinggi	rendah	rendah	tinggi	tinggi
Keleluasaan pengguna	rendah	menengah	menengah	menengah	tinggi
Media penjelasan	Suara & bahasa tubuh	suara	suara	Teks, gambar, suara, video	Teks, gambar, suara, video
Tersedia informasi bagi tuna netra	ya	ya	ya	ya	
Tersedia informasi bagi tuna rungu	Tidak begitu baik	tidak	tidak	ya	ya
Berbahasa asing	Tidak begitu baik	ya	ya	ya	ya

Pada tabel hasil analisa tersebut tampak bahwa pada perangkat hasil pembuatan tugas akhir ini didapatkan sistem pemandu wisata yang memiliki beberapa keuggulan dibandingkan dengan perangkat lainnya, yaitu rendahnya biaya untuk melakukan perubahan konten, tingginya nilai keleluasaan pengguna, kesesuaian konten secara otomatis, media penjelasan yang lebih legkap dan menarik, serta kemampuan menampilkan informasi dalam berbagai berbahasa.

BAB 5
PENUTUP

5.1. Kesimpulan

Dari pengujian dan analisa pembuatan tugas akhir ini dapat diambil beberapa kesimpulan yaitu:

- Proses pendeteksian gambar marker lebih optimal pada intensitas cahaya yang besar.
- Perangkat pada tugas akhir ini dapat mendeteksi gambar dengan jarak efektif 0.5 meter hingga 1.5 meter dari gambar marker.
- Untuk mendeteksi gambar marker, kinerja perangkat lebih optimal apabila diletakkan pada sudut efektif sekitar 70 hingga -70 derajat dari gambar marker.
- Proses pengolahan citra OpenCV pada Raspberry Pi pada tugas akhir ini memiliki kecepatan maksimal hingga 7fps. Kecepatan ini kurang mencukupi untuk dapat menampilkan video informasi secara halus.

5.2. Saran

- Untuk mempercepat kinerja perangkat, dapat digunakan *procecing unit* jenis lain yang memiliki spesifikasi performa yang lebih baik.
- Hasil perangkat HMD pada tugas akhir ini masih terlalu besar untuk dipakai. Untuk mendapatkan bentuk HMD yang lebih minimalis, dapat digunakan lensa konveks-konkav dengan titik focus yang lebih kecil, dan layar LCD dengan dimensi yang lebih kecil juga.
- Hasil video yang ditampilkan pada perangkat tersebut sedikit pecah dikarenakan layar LCD yang digunakan memiliki resolusi yang kecil. Untuk mendapatkan hasil video yang maksimal, dapat digunakan layar LCD dengan resolusi yang lebih besar.

DAFTAR PUSTAKA

[1] Anonim, “Kamus Besar Bahasa Indonesia (KBBI) Online” <URL: http://kbbi.web.id/pramuwisata>, Mei, 2015.

[2] Anonim, “Situs Resmi Pemerintah Kota Surabaya” <URL: http://www.surabaya.go.id/dinamis/?id=583>, Mei, 2015.

[3] Anonim, "OpenCV" <URL:http://OpenCV.org/about.html> Mei, 2015.

[4] Anonim, “Raspberry Pi - Teach, Learn, and Make with Raspberry Pi" <URL: https://www.raspberrypi.org/help/what-is-a-Raspberry-pi/> Mei, 2015.

[5] Anonim, “OMXPlayer Builds" <URL: http://omxplayer.sconde.net/>, Mei, 2015.

[6] Bretschneider, Nora dkk, "*Head Mounted Displays for Fire Fighters*", in Applied Wearable Computing (IFAWC), 2006 3rd International Forum on, Bremen, Germany, 2006.

[7] Thomas, B.H. dan Sandor Christian, "*What Wearable Augmented reality Can Do for You*", Pervasive Computing, IEEE, vol. 8, no. 2, pp. 8-11, 2009.

[8] K. Kiyokawa, "*Trends and Vision of Head Mounted Display in Augmented reality*", in Ubiquitous Virtual Reality (ISUVR), 2012 International Symposium on, Adaejeon, 2012.

[6] Chou, Li-Der dkk, "*Requirement Analysis and Implementation of Palm-Based Multimedia Museum Guide Systems*", in Advanced Information Networking and Applications, 2004 18th International Conference on, Fukuoka, Japan, 2004.

[7] Anonim, “Logitech Webcam C170" <URL: http://www.logitech.com/id-id/product/webcam-c170>, Mei, 2015.

LAMPIRAN

Program OpenCV

```
#ifdef _CH_
#pragma package <opencv>
#endif

#define CV_NO_BACKWARD_COMPATIBILITY

#include <opencv/cv.h>
#include <opencv/highgui.h>
#include <stdio.h>
#include <math.h>
#include <string.h>

CvSize extraDisplay = cvSize(0, 0);
CvSize ukuranawal = cvSize(320, 240);
CvSize ukuranakhir = cvSize(668, 530);
int thresh = 50;
IplImage* img = 0;
IplImage* img1 = 0;
IplImage* img2 = cvCreateImage(ukuranakhir, 8, 3);
IplImage* img3 = cvCreateImage(cvSize(640, 480), 8,
3);
CvMemStorage* storage = 0;
int cv_scanMarker(IplImage* sourceImage, CvPoint
titik[4]);
IplImage* displayout = cvCreateImage(cvSize(640,
480), 8, 3);
CvCapture* display_out1 = 0;
CvPoint2D32f dispQuad[4], srcQuad[4], dstQuad[4],
imgQuad[4], imgQuad2[4];
CvPoint MouseFilter(CvPoint mop);
```

```
CvPoint mousep[7], jarak[7];
CvPoint titiktampil = cvPoint(100, 100);
int tempx1, tempx2, tempxa, tempxb, tempya, tempyb,
jarakx, jaraky, awalx, awaly = 0;
int tampil = 0;


double angle(CvPoint* pt1, CvPoint* pt2, CvPoint*
pt0);
CvSeq* findSquares4(IplImage* img, CvMemStorage*
storage);
void drawSquares(IplImage* img, CvSeq* squares);
void cv_ARaugmentImage(IplImage* display, IplImage*
img, CvPoint koordinat, double scale);
CvPoint JarakFilter(CvPoint ukuran);
int marker_id = 0;
int statevideo = 0;
double t1, t2, t3 = 0;


int main(int argc, char** argv)
{
	int i, c, r;
	storage = cvCreateMemStorage(0);
	CvCapture* capture = 0;
	capture = cvCaptureFromCAM(0);
	for(r=0;r<200;r++)
	{
		t1 = cvGetTickCount();
		img1 = cvQueryFrame(capture);
		if (!img1)
		{
			return -1;
```

```
            }

            t1 = cvGetTickCount() - t1;
            t2 = cvGetTickCount();
            img = cvCreateImage(ukuranawal, 8, 3);
            cvResize(img1, img, 1);
            drawSquares(img, findSquares4(img,
storage));
            cvClearMemStorage(storage);
            c = cvWaitKey(1);
            t3 = cvGetTickCount() - t3;
            printf(" Cam %.4gms Proc %.4gms Omx
%.4gms : FPS = %.4g\n", t1 /
((double)cvGetTickFrequency()*1000.), t2 /
((double)cvGetTickFrequency()*1000.), t3 /
((double)cvGetTickFrequency()*1000.),
((double)cvGetTickFrequency()*1000.*1000.) / (t1 + t2
+ t3));
      }
      cvReleaseCapture(&capture);
      cvReleaseImage(&img1);
      cvReleaseImage(&img);
      return 0;
}

double angle(CvPoint* pt1, CvPoint* pt2, CvPoint*
pt0)
{
      double dx1 = pt1->x - pt0->x;
      double dy1 = pt1->y - pt0->y;
      double dx2 = pt2->x - pt0->x;
      double dy2 = pt2->y - pt0->y;
```

```
	return (dx1*dx2 + dy1*dy2) / sqrt((dx1*dx1 +
dy1*dy1)*(dx2*dx2 + dy2*dy2) + 1e-10);
}

CvSeq* findSquares4(IplImage* img, CvMemStorage*
storage)
{
	CvSeq* contours;
	int i, c, l, N = 11;
	CvSize sz = cvSize(img->width & -2, img-
>height & -2);
	IplImage* timg = cvCloneImage(img);
	IplImage* gray = cvCreateImage(sz, 8, 1);
	IplImage* pyr = cvCreateImage(cvSize(sz.width
/ 2, sz.height / 2), 8, 3);
	IplImage* tgray;
	CvSeq* result;
	double s, t;
	CvSeq* squares = cvCreateSeq(0, sizeof(CvSeq),
sizeof(CvPoint), storage);
	cvSetImageROI(timg, cvRect(0, 0, sz.width,
sz.height));
	cvPyrDown(timg, pyr, 7);
	cvPyrUp(pyr, timg, 7);
	tgray = cvCreateImage(sz, 8, 1);
	for (c = 0; c < 1; c++)
	{
		for (l = 0; l < 1; l++)
		{
			cvCvtColor(timg, tgray,
CV_RGB2GRAY);
			cvThreshold(tgray, gray, 1, 255,
CV_THRESH_OTSU);
```

```
                    tgray = cvCloneImage(gray);
                    cvFindContours(gray, storage,
&contours, sizeof(CvContour),
                           CV_RETR_LIST,
CV_CHAIN_APPROX_SIMPLE, cvPoint(0, 0));
                    while (contours)
                    {
                           result =
cvApproxPoly(contours, sizeof(CvContour), storage,
                                  CV_POLY_APPROX_DP,
cvContourPerimeter(contours)*0.02, 0);
                           if (result->total == 4 &&

      cvContourArea(result, CV_WHOLE_SEQ, 0) > 1000
&&

      cvCheckContourConvexity(result))
                           {
                                  s = 0;
                                  for (i = 0; i < 5;
i++)
                                  {
                                         if (i >= 2)
                                         {
                                                t =
fabs(angle(

      (CvPoint*)cvGetSeqElem(result, i),

      (CvPoint*)cvGetSeqElem(result, i - 2),

      (CvPoint*)cvGetSeqElem(result, i - 1)));
```

```
                                                    s = s >
t ? s : t;
                                                }
                                        }
                                        if (s < 0.3)
                                        for (i = 0; i < 4;
i++)

        cvSeqPush(squares,

        (CvPoint*)cvGetSeqElem(result, i));
                                }
                                contours = contours-
>h_next;
                        }
                }
        }
        t2 = cvGetTickCount() - t2;
        t3 = cvGetTickCount();
        CvSeqReader reader;
        cvStartReadSeq(squares, &reader, 0);
        for (i = 0; i < squares->total; i += 4)
        {
                CvPoint pt[4], *rect = pt;
                int count = 4;
                CV_READ_SEQ_ELEM(pt[0], reader);
                CV_READ_SEQ_ELEM(pt[1], reader);
                CV_READ_SEQ_ELEM(pt[2], reader);
                CV_READ_SEQ_ELEM(pt[3], reader);
                marker_id = cv_scanMarker(tgray, pt);
                printf("code = %2d", marker_id);
        }
        cvSetZero(img2);
```

```
	if (tampil == 50)
	{
		char perintah[50];
		sprintf(perintah, "/home/pi/coba/dbus.sh
setvideopos '%d %d %d %d'", titiktampil.x-600,
titiktampil.y-500, titiktampil.x+600,
titiktampil.y+500);
		system(perintah);
		tampil--;
	}

	else if (tampil > 30)
	{
		tampil--;
	}

	else if (tampil == 30)
	{
		statevideo = 0;
		system("/home/pi/coba/dbus.sh
setposition 0");
		system("/home/pi/coba/dbus.sh
hidevideo");
		tampil--;
	}

	else if (tampil > 0)
	{
		tampil--;
	}

	else
	{
```

```
            tampil = 30;
      }

      cvReleaseImage(&gray);
      cvReleaseImage(&pyr);
      cvReleaseImage(&tgray);
      cvReleaseImage(&timg);
      return squares;
}

void drawSquares(IplImage* img, CvSeq* squares)
{
      CvSeqReader reader;
      IplImage* cpy = cvCloneImage(img);
      int i;

      cvStartReadSeq(squares, &reader, 0);

      for (i = 0; i < squares->total; i += 4)
      {
            CvPoint pt[4], *rect = pt;
            int count = 4;
            CV_READ_SEQ_ELEM(pt[0], reader);
            CV_READ_SEQ_ELEM(pt[1], reader);
            CV_READ_SEQ_ELEM(pt[2], reader);
            CV_READ_SEQ_ELEM(pt[3], reader);
      }
      cvReleaseImage(&cpy);
}

int cv_scanMarker(IplImage* sourceImage, CvPoint
titik[4])
{
```

```
	CvMat* warp_matrix = cvCreateMat(3, 3,
CV_32FC1);
	int marker_id;
	int appx, appy = 0;
	IplImage* marker_transposed_img =
cvCreateImage(cvSize(180, 180), 8, 1);

	srcQuad[0].x = titik[1].x;
	srcQuad[0].y = titik[1].y;
	srcQuad[1].x = titik[0].x;
	srcQuad[1].y = titik[0].y;
	srcQuad[2].x = titik[2].x;
	srcQuad[2].y = titik[2].y;
	srcQuad[3].x = titik[3].x;
	srcQuad[3].y = titik[3].y;

	dstQuad[0].x = 0;
	dstQuad[0].y = 0;
	dstQuad[1].x = 180;
	dstQuad[1].y = 0;
	dstQuad[2].x = 0;
	dstQuad[2].y = 180;
	dstQuad[3].x = 180;
	dstQuad[3].y = 180;

	cvGetPerspectiveTransform(srcQuad, dstQuad,
warp_matrix);
	cvWarpPerspective(sourceImage,
marker_transposed_img, warp_matrix);
	int value = 0;
	int i = 0;
	for (int y = 45; y < 160; y += 30)
	{
```

```
            for (int x = 45; x < 160; x += 30)
            {
                  uchar* pointer_scanline =
(uchar*)(marker_transposed_img->imageData + (y -
1)*marker_transposed_img->width);
                  if (pointer_scanline[x - 1] == 0)
                  {
                        value += (int)pow(2, i);
                  }
                  i++;
                  cvCircle(marker_transposed_img,
cvPoint(x, y), 1, CV_RGB(0, 0, 255), 6, 8, 0);
            }
      }
      marker_id = value;
      if (marker_id == 42405 || marker_id == 23130
|| marker_id == 39498 || marker_id == 34093 ||
marker_id == 9252 || marker_id == 2640 )
      {
            tempx1 = abs(titik[0].x - titik[1].x);
            tempx2 = abs(titik[1].x - titik[2].x);
            tempxa = abs(titik[0].x - titik[2].x);
            tempxb = abs(titik[1].x - titik[3].x);
            tempya = abs(titik[0].y - titik[2].y);
            tempyb = abs(titik[1].y - titik[3].y);
            if (tempx1 > tempx2)
            {
                  if (tempxa > tempxb)
                  {
                        jarakx = tempxa;
                        jaraky = tempya;
                  }
                  else
```

```
		{
			jarakx = tempxb;
			jaraky = tempxb;
		}

		if (titik[0].x < titik[1].x)
			awalx = titik[0].x;
		else
			awalx = titik[1].x;
		if (titik[1].y < titik[2].y)
			awaly = titik[1].y;
		else
			awaly = titik[2].y;
	}
	else
	{
		if (tempxa > tempxb)
		{
			jarakx = tempxa;
			jaraky = tempya;
		}
		else
		{
			jarakx = tempxb;
			jaraky = tempxb;
		}

		if (titik[1].x < titik[2].x)
			awalx = titik[1].x;
		else
			awalx = titik[2].x;
		if (titik[0].y < titik[1].y)
			awaly = titik[0].y;
```

```
                        else
                                awaly = titik[1].y;
                }
                titiktampil = cvPoint((awalx + (jarakx /
2))*ukuranakhir.width / ukuranawal.width, (awaly +
(jaraky / 2))*ukuranakhir.height /
ukuranawal.height);
                titiktampil.x = ukuranakhir.width -
titiktampil.x;
                printf(" posisi %d %d ",
titiktampil.x,titiktampil.y);
                if (marker_id == 42405 || marker_id ==
23130)
                {
                        if (titiktampil.x <
ukuranakhir.width/2 - 150 && statevideo != 1)
                        {
                                printf("kanan");

        system("/home/pi/coba/dbus.sh setposition
180000000");
                                statevideo = 1;
                        }
                        else if (titiktampil.y >
ukuranakhir.height/2 + 100 && statevideo != 2)
                        {
                                printf("atas");

        system("/home/pi/coba/dbus.sh setposition
360000000");
                                statevideo = 2;
                        }
```

```
                    else if (titiktampil.x >
ukuranakhir.width/2 + 150 && statevideo != 3)
                    {
                           printf("kiri");

       system("/home/pi/coba/dbus.sh setposition
540000000");
                           statevideo = 3;
                    }
                    else if (titiktampil.y <
ukuranakhir.height/2 - 100 && statevideo != 4)
                    {
                           printf("bawah");

       system("/home/pi/coba/dbus.sh setposition
720000000");
                           statevideo = 4;
                    }
                    else if (titiktampil.x >
ukuranakhir.width/2 - 100 && titiktampil.x <
ukuranakhir.width/2 + 100 && titiktampil.y >
ukuranakhir.height/2 - 50 && titiktampil.y <
ukuranakhir.height/2 + 50 && statevideo !=5 )
                    {
                           printf("tengah");

       system("/home/pi/coba/dbus.sh setposition
70000000");
                           statevideo = 5;
                    }
             }
```

```
            else if(marker_id == 39498 || marker_id
== 34093)
            {
                  if (statevideo != 6)
                  {

      system("/home/pi/coba/dbus.sh setposition
900000000");
                        statevideo = 6;
                  }
            }

            else if(marker_id == 9252 || marker_id
== 2640)
            {
                  if(statevideo != 7)
                  {

      system("/home/pi/coba/dbus.sh setposition
1200000000");
                        statevideo = 7;
                  }
            }

            if (tampil < 30)
            {
                  system("/home/pi/coba/dbus.sh
unhidevideo");
                  tampil = 50;
            }

            else if (tampil < 49)
            {
```

```
                    tampil = 50;
                }
        }
        cvReleaseMat(&warp_matrix);
        cvReleaseImage(&marker_transposed_img);
        return marker_id;
}

CvPoint MouseFilter(CvPoint mop)
{
        mousep[6] = mousep[5];
        mousep[5] = mousep[4];
        mousep[4] = mousep[3];
        mousep[3] = mousep[2];
        mousep[2] = mousep[1];
        mousep[1] = mousep[0];
        mousep[0] = mop;
        int bufx = 0, bufy = 0;
        for (int i = 0; i<7; i++)
        {
                bufx = bufx + mousep[i].x;
                bufy = bufy + mousep[i].y;
        }
        bufx = bufx / 7;
        bufy = bufy / 7;
        return cvPoint(bufx, bufy);
}

CvPoint JarakFilter(CvPoint ukuran)
{
        jarak[6] = jarak[5];
        jarak[5] = jarak[4];
        jarak[4] = jarak[3];
```

```
        jarak[3] = jarak[2];
        jarak[2] = jarak[1];
        jarak[1] = jarak[0];
        jarak[0] = ukuran;
        int bufx2 = 0, bufy2 = 0;
        for (int i = 0; i<7; i++)
        {
               bufx2 = bufx2 + jarak[i].x;
               bufy2 = bufy2 + jarak[i].y;
        }
        bufx2 = bufx2 / 7;
        bufy2 = bufy2 / 7;
        return cvPoint(bufx2, bufy2);
}
```

Program Dbus Connection

```
#!/BIN/BASH

#SET -X

OMXPLAYER_DBUS_ADDR="/TMP/OMXPLAYERDBUS.${USER}"
OMXPLAYER_DBUS_PID="/TMP/OMXPLAYERDBUS.${USER}.PID"
EXPORT                    DBUS_SESSION_BUS_ADDRESS=`CAT
$OMXPLAYER_DBUS_ADDR`
EXPORT DBUS_SESSION_BUS_PID=`CAT $OMXPLAYER_DBUS_PID`

[  -Z  "$DBUS_SESSION_BUS_ADDRESS"  ]  &&  {  ECHO  "MUST
HAVE DBUS_SESSION_BUS_ADDRESS" >&2; EXIT 1; }

CASE $1 IN
STATUS)
      DURATION=`DBUS-SEND  --PRINT-REPLY=LITERAL  --
SESSION            --REPLY-TIMEOUT=500            --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.FREEDESKTOP.DBUS.PROPERTIES.DURATION`
      [ $? -NE 0 ] && EXIT 1
      DURATION="$(AWK '{PRINT $2}' <<< "$DURATION")"

      POSITION=`DBUS-SEND  --PRINT-REPLY=LITERAL  --
SESSION            --REPLY-TIMEOUT=500            --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.FREEDESKTOP.DBUS.PROPERTIES.POSITION`
      [ $? -NE 0 ] && EXIT 1
      POSITION="$(AWK '{PRINT $2}' <<< "$POSITION")"
```

```
	PLAYSTATUS=`DBUS-SEND --PRINT-REPLY=LITERAL --
SESSION		--REPLY-TIMEOUT=500		--
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.FREEDESKTOP.DBUS.PROPERTIES.PLAYBACKSTATUS`
	[ $? -NE 0 ] && EXIT 1
	PLAYSTATUS="$(SED 'S/^ *//;S/ *$//;' <<<
"$PLAYSTATUS")"

	PAUSED="TRUE"
	[ "$PLAYSTATUS" == "PLAYING" ] &&
PAUSED="FALSE"
	ECHO "DURATION: $DURATION"
	ECHO "POSITION: $POSITION"
	ECHO "PAUSED: $PAUSED"
	;;

VOLUME)
	VOLUME=`DBUS-SEND --PRINT-REPLY=DOUBLE --
SESSION		--REPLY-TIMEOUT=500		--
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.FREEDESKTOP.DBUS.PROPERTIES.VOLUME
${2:+DOUBLE:}$2`
	[ $? -NE 0 ] && EXIT 1
	VOLUME="$(AWK '{PRINT $2}' <<< "$VOLUME")"
	ECHO "VOLUME: $VOLUME"
	;;

PAUSE)
	DBUS-SEND --PRINT-REPLY=LITERAL --SESSION --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
```

```
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:16
>/DEV/NULL
      ;;

STOP)
      DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:15
>/DEV/NULL
      ;;

SEEK)
      DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.SEEK           INT64:$2
>/DEV/NULL
      ;;

SETPOSITION)
      DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.SETPOSITION
OBJPATH:/NOT/USED INT64:$2 >/DEV/NULL
      ;;

SETALPHA)
      DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
```

```
ORG.MPRIS.MEDIAPLAYER2.PLAYER.SETALPHA
OBJPATH:/NOT/USED INT64:$2 >/DEV/NULL
       ;;

SETVIDEOPOS)
       DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.VIDEOPOS
OBJPATH:/NOT/USED STRING:"$2 $3 $4 $5" >/DEV/NULL
       ;;

HIDEVIDEO)
       DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:28
>/DEV/NULL
       ;;

UNHIDEVIDEO)
       DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:29
>/DEV/NULL
       ;;

VOLUMEUP)
       DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
```

```
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:18
>/DEV/NULL
      ;;

VOLUMEDOWN)
      DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:17
>/DEV/NULL
      ;;

TOGGLESUBTITLES)
      DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:12
>/DEV/NULL
      ;;

HIDESUBTITLES)
      DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:30
>/DEV/NULL
      ;;

SHOWSUBTITLES)
      DBUS-SEND  --PRINT-REPLY=LITERAL  --SESSION  --
DEST=ORG.MPRIS.MEDIAPLAYER2.OMXPLAYER
/ORG/MPRIS/MEDIAPLAYER2
```

```
ORG.MPRIS.MEDIAPLAYER2.PLAYER.ACTION         INT32:31
>/DEV/NULL
      ;;

*)
      ECHO                 "USAGE:                 $0
STATUS|PAUSE|STOP|SEEK|VOLUMEUP|VOLUMEDOWN|SETPOSITIO
N                    [POSITION                    IN
MICROSECONDS]|HIDEVIDEO|UNHIDEVIDEO|TOGGLESUBTITLES|H
IDESUBTITLES|SHOWSUBTITLES|SETVIDEOPOS   [X1   Y1   X2
Y2]|SETALPHA [ALPHA (0..255)]" >&2
      EXIT 1
      ;;
ESAC
```

RIWAYAT HIDUP PENULIS

Reza Zhafiri dilahirkan di Surabaya, 3 April 1993. Merupakan putra ke pertama dari tiga bersaudara. Penulis menempuh jenjang pendidikan dari SD Ta'miriyah Surabaya pada 1999/2005, SMP Negeri 2 Surabaya pada 2005/2008, dan SMA Negeri 5 Surabaya pada 2008/2011. Pada tahun 2011, penulis melanjutkan studinya ke Institut Teknologi Sepuluh Nopember Surabaya, Fakultas Teknologi Industri, Jurusan Teknik Elektro dan mengambil bidang studi Elektronika Industri. Penulis banyak melakukan aktivitas pada bidang keilmiahan maupun organisasi. Sejak tahun 2011, penulis berhasil menjuarai berbagai kompetisi di bidang keilmiahan. Beberapa diantaranya yaitu Pekan Ilmiah Mahasiswa Nasional selama 3 tahun berturut-turut pada tahun 2011-2013 dan Go Green in The City sebagai Runner-Up pada tahun 2013. Dalam bidang organisasi, penulis aktif dalam Workshop Jurusan Teknik Elektro dan menjadi ketua Workshop pada tahun 2013/2014. Hobi dalam bidang robotika juga dikembangkan oleh penulis. Pada tahun 2011-2013 penulis tercatat sebagai pembimbing ekstrakurikuler robotika SMA Negeri 5 Surabaya. Hal ini juga ditunjukkan pada BARONAS 2012-2014, penulis merupakan juri pada kompetisi line-tracer tingkat nasional. Penulis dapat dihubungi melalui email zazhaf@gmail.com